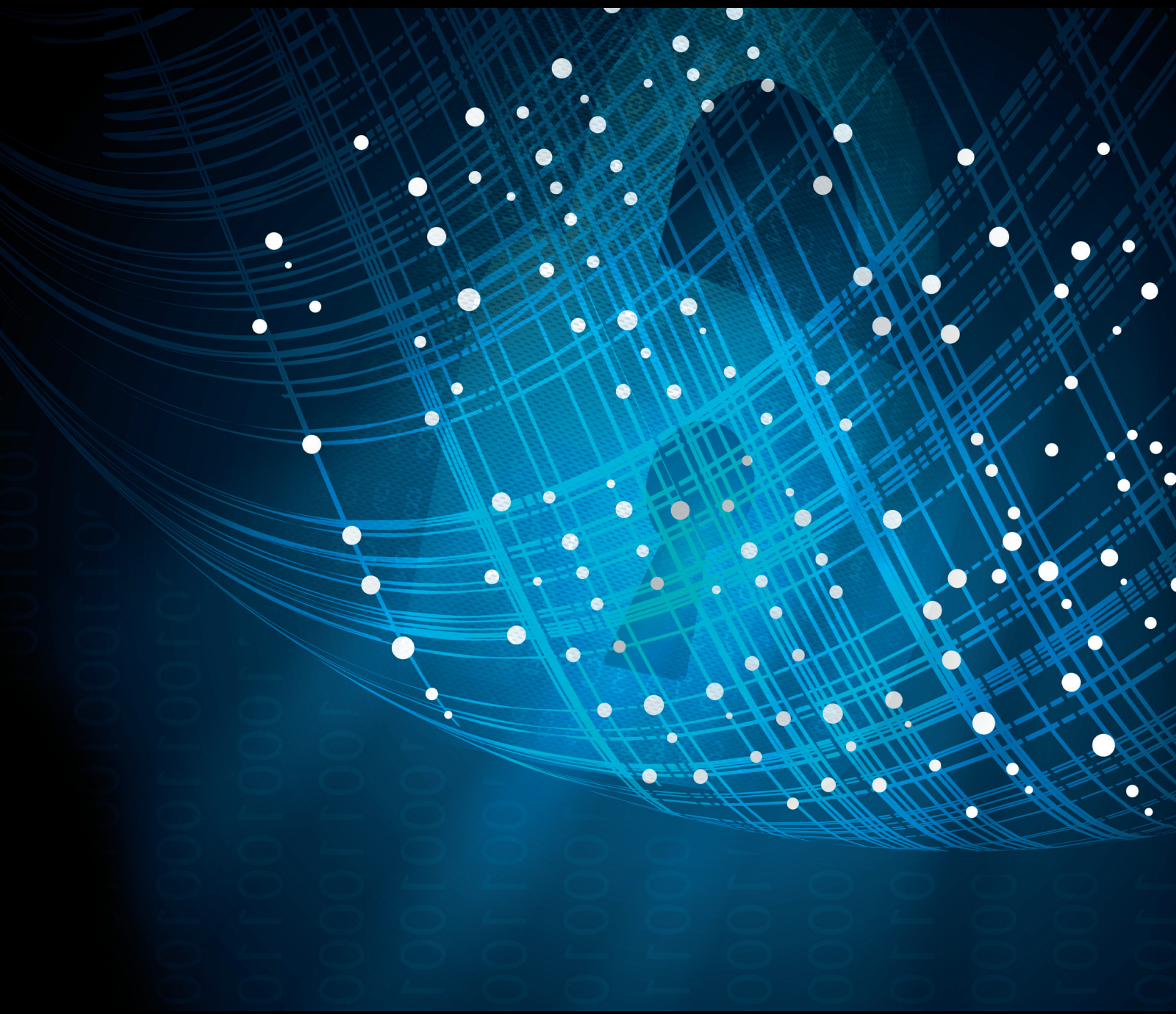
Security and Communication Networks

Multimodality Data Analysis in Information Security

Lead Guest Editor: Liguo Zhang
Guest Editors: Wei Ke and Xinguang Xiang



Multimodality Data Analysis in Information Security

Security and Communication Networks

Multimodality Data Analysis in Information Security

Lead Guest Editor: Liguo Zhang
Guest Editors: Wei Ke and Xinguang Xiang

Contents

Contents

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9975912, 9 pages
https://doi.org/10.1155/2021/9975912

Research Article

A Data Aggregation Privacy Protection Algorithm Based on Fat Tree in Wireless Sensor Networks

Cheng Li,[1,2] **Guoyin Zhang**,[1] **Yan Mao**,[1] **and Xin Zhao**[1]

[1]*College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China*
[2]*College of Computer and Control Engineering, Qiqihar University, Qiqihar 161006, China*

Correspondence should be addressed to Guoyin Zhang; zhangguoyin@hrbeu.edu.cn

Received 12 March 2021; Accepted 11 August 2021; Published 13 September 2021

Academic Editor: Mohamed Amine Ferrag

Wireless sensor network is the momentous part of the Internet of Things. Data aggregation technology is the most practical method to reduce the amount of communication among nodes. Adding a privacy protection mechanism to data aggregation is one of the important means for privacy protection and security in wireless sensor networks. Aiming at certain performance defects of the existing SMART (Slice-Mix-AggRegaTe) privacy protection algorithms, a fat tree-based data aggregation privacy protection algorithm is proposed in this paper, which is referred to as the FTSMART (Fat Tree Slice-Mix-AggRegaTe) algorithm. Concerning the innovative algorithm, the fat tree (FT) is introduced, the fat tree structure is adopted to optimize the data slicing scheme and the aggregation tree generation scheme, and the allocation of fixed time intervals is employed for nodes to reduce the transmission collision in the data aggregation process and to guarantee the completion of the data transmission. The simulation experiment results demonstrate that the FTSMART algorithm has presented the favorable performance in terms of the privacy protection, the network communication overhead, and the data aggregation accuracy.

1. Introduction

Wireless sensor network (WSN) is the multihop and self-organized networks formed from a large amount of wireless sensor nodes by wireless communications. A large number of sensor nodes are randomly deployed in various environments (such as harsh environments where humans cannot stay for a long time) to perceive and collect the target information so that people can analyze and process the information and make the reasonable judgments. Currently, wireless sensor network has been widely applied in the environmental monitoring, the military fields, the intelligent transportation, the logistics tracking, the intelligent medical, and so forth. According to the energy-constrained characteristics of wireless sensor network, the data aggregation technology is widely used. The core of the data aggregation technology is to aggregate the data from different information sources, remove the redundant information, and reduce the amount of transmitted data by means of the data compression and the feature extraction. Accordingly, the network energy consumption is decreased, the network life cycle is extended, and the efficiency and accuracy of data collection are improved greatly [1].

Wireless sensor network has a low security to some certain. In the meantime, the data privacy protection mechanism is not provided by the basic data aggregation technology generally. However, in practical applications, due to the characteristics of the wireless transmission, the data transmitted among nodes is easy to be captured and eavesdropped, and the data of the child nodes is obtained by the trusted parent nodes in networks. If the wireless link is broken or the parent nodes are captured by the attacker, the private data is exposed accordingly. The data aggregation privacy protection technology of wireless sensor network is the technology that prevents the private data from being acquired even if the transmitted data is captured and decrypted externally or captured by other trusted nodes internally, under the condition that the data aggregation result is correct. So far, in the previous researches, some privacy protection schemes have been proposed, each of

which has its own scope of application, and some of the schemes still have some problems that need to be further resolved.

Based on the research of the data slicing aggregation in the SMART (Slice-Mix-AggRegaTe) algorithm [2], the fat tree [3] is introduced into the domain of wireless sensor network, and a fat tree-based data aggregation privacy protection algorithm is proposed in this paper. An aggregation tree is constructed through the fat tree, a hop-by-hop data aggregation method is adopted, and the data slicing technology is used for the privacy protection. The main contributions of the paper are as follows:

(1) The fat tree structure is introduced to construct a fusion tree. The structure and characteristics of the fat tree are fully utilized, and the constructed fusion tree is the shortest path tree, which realizes the advantages of low computational complexity, less data transmission, and small fusion delay.

(2) The fat tree structure is combined with the data slicing technology to protect data privacy. The data is sliced according to the number of parent nodes in the fat tree where the fusion tree node is located, and the sensing data of all the nodes is sliced and fused, which improves the performance of data privacy protection.

(3) The fixed time intervals (time slices) are allocated to complete the data fusion scheduling. The process of generating the fusion tree from the fat tree is carried out layer by layer from the leaf node to the root node. At the same time, a fixed time interval can be allocated to each node according to the scheduling principles, so as to avoid the data transmission collision among nodes and improve the accuracy of fusion.

The remainder of the paper is organized as follows. Section 2 reviews the previous works. In Section 3, we described the FTSMART algorithm, including the system model and the specific implementation process in detail. In Section 4, the performance of the FTSMART algorithm is evaluated by the simulation experiments, covering the privacy protection and the communication overhead as well as the aggregation accuracy. Section 5 concludes the research and the future work.

2. Previous Works

He et al. [2] proposed the Privacy-preserving Data Aggregation (PDA) algorithm and researched the additive aggregation function SUM, which included the Cluster-based Private Data Aggregation (CPDA) algorithm and the SMART algorithm. But the CPDA algorithm was the data aggregation algorithm based on clustering, and its calculation process was complicated and computationally expensive. The SMART algorithm is an algorithm closely related to this paper. Its central thoughts are to cut the acquired data into several fragments (i.e., slices), and to transmit the sliced data along different transmission paths. In this way, unless all the slices are obtained by the privacy attacker, the ultimate private information cannot be obtained. The specific implementation process of the SMART algorithm is divided into 3 steps:

In the first phase (slicing), the collected data is randomly sliced into J slices by each node, 1 slice is kept by itself, and the remaining (J-1) data slices are encrypted and randomly sent to the neighboring nodes.

In the second phase (mixing), the intermediate node waits for a period of time to receive the data slices sent by other nodes. When the intermediate node receives the encrypted data slice, the shared key is used to perform decryption, and then all the data slices perform the mixed calculation.

In the final phase (aggregating), all the nodes employ the data aggregation tree that are established by the Tiny Aggregation service for ad hoc sensor networks (TAG) algorithm [4] to transfer up the data aggregation results layer by layer.

The advantages of the SMART algorithm are that each node slices its own private information and mixes the data slices of different nodes, which increases the difficulty for an attacker to eavesdrop the complete data multiplication. It can be indicated that the algorithm has the prominent performance of the data privacy protection as well as the less computational overhead. The disadvantages of the SMART algorithm are that, with the high network communication overhead, the number of data packets generated is several times that of many other algorithms, and it is not suitable for the networks with a large number of nodes and the densely distributed networks. Otherwise, the amount of the sliced data is larger, which not only affects the normal working efficiency of the networks but also seriously affects the accuracy of the data aggregation.

A great amount of research studies have been carried out on the SMART algorithm by many scholars, and plenty of improvements have been presented [5–13]. The study in [5] proposed the EEHA algorithm, which only permitted the data collected by the leaf nodes of the aggregation tree to be sliced, so that the number of slices was reduced, and the network communication overhead was reduced, but the data privacy protection performance was reduced as well. The study in [6] presented the ESMART algorithm on the basis of [5]. The number of the leaf node data slices was random between [2, J] (J is the maximum number of slices that were set), which improved the data privacy protection performance greatly. In [7], the HEEPP algorithm was proposed on the basis of [6], which added a data query mechanism, queried the data transmission status of the subnodes when idle, prevented the data loss, and thus improved the aggregation accuracy. In [8], in order to improve the accuracy of the aggregation, the LTPART algorithm was presented. In the data aggregation phase, a fixed time slice or a floating time slice was allocated to each layer of nodes to guarantee the full aggregation. In [6] and [7], there existed the problem that the number of the random slices was larger than the number of the neighboring nodes, so the slicing data could not be exchanged and mixed normally. In order to solve the problem, the study in [9] introduced the SESDA algorithm,

and the number of packets was dynamically determined by the number of the data slices that the nodes had received. The ESPART algorithm proposed in [10] reduced the data packets generated by segmentation and reassembly by controlling the inside and outside of nodes. The study in [11] put forward the PSMART algorithm and added a local slicing strategy. The nodes that failed to be sliced have not undergone slicing, but end-to-end encryption. In order to further improve the performance of the data privacy protection, the work in [12] proposed the D-SMART algorithm, and the data was divided according to the deviation degree, the ordinary data was cut into 2 slices, the important data was cut into 3 slices, and the confidential data was cut into 4 slices. Aiming at the data collision problem caused by the segmentation and recombination technology, the work in [13] introduced five optimization factors to reduce the collision rate and reduce the loss caused by collision. Some scholars have proposed various related applications based on the SMART algorithm, which have been illustrated in [14, 15] as well.

On the basis of the related researches above, an aggregation tree that is suitable for the fragmented data aggregation is constructed to research the privacy protection algorithm in the paper, and the SMART algorithm and its relative algorithms have been improved to a certain extent. Meanwhile, the energy consumption overhead performance of network communications is taken into account.

3. Data Aggregation Privacy Protection Algorithm Based on Fat Tree

The fat tree-based data aggregation privacy protection algorithm is proposed in the paper, which is composed of four steps: the fat tree construction phase, the slicing phase, the mixing phase, and the aggregating phase. In this section, the system model, the implementation process, and the advantages of the proposed algorithm are elaborated and illustrated in detail.

3.1. System Model. In a two-dimensional square area of $L \times L$, there are N nodes randomly distributed, and only one base station is considered, namely, the Sink node, which constitutes the extremely connected wireless sensor network $G(V, E)$ with Sink. All the nodes in the network G receive and send data and have the same transmission radius r. In the research on the problem, the following conditions are set:

(1) All the nodes in the network have completed the data perception.

(2) The data aggregation that occurs in the network is the complete aggregation; that is, no matter how many information is received by node i, the node encapsulates all the information in a data packet for transmission. The data aggregation function is defined [16, 17], which is shown in

$$y(t) = f\left(d_1(t) + d_2(t) + \cdots + d_i(t)\right), \quad (i = 1, 2, \cdots, N), \tag{1}$$

where $d_i(t)$ represents the data collected by node i at time t. The research in the paper supports the additive aggregation calculation, such as summation, expectation, or variance. A typical SUM aggregation function is shown in

$$y(t) = \sum_{i=1}^{N} d_i(t). \tag{2}$$

According to formula (2), the aggregation process using the SUM function is shown in Figure 1.

(3) It is agreed upon that any node in the network sends and receives the data within 1 hop, but sending data and receiving data are nonparallel, and receiving two or more data packets at the same time is nonparallel as well.

(4) All the transmissions that can be carried out at the same time (no collision between each other) are completed in a unit of time, which is recorded as a time interval. The internode transmissions have the uniform time intervals.

(5) As with the SMART algorithm, a random key distribution mechanism is adopted to encrypt and decrypt the transmitted data in the paper and complete the communications among nodes through a shared key [2, 12]. The probability that two random nodes in the network have the same key is expressed as follows:

$$Peavesdrop = \frac{k}{K}, \tag{3}$$

where K is the total number of keys in the key pool and k is the number of keys in the key ring. *Peavesdrop* represents the probability that the links between any pair of communication nodes can be cracked by the attacker, that is, the probability that the private data is eavesdropped on. When the key pool is large enough, the security of the mechanism is stronger.

3.2. Fat Tree Construction Phase. The purpose of constructing the fat tree is to serve as the basis of node data slicing, finally cutting it into the shortest path aggregation tree. Therefore, the characteristics of the fat tree are listed as follows: there is a unique root node, namely, the Sink node, to ensure that the data is finally gathered on the Sink node; the distance between any node and its parent nodes (except the Sink node) or between any node and its child nodes (except for the leaf nodes) must be less than or equal to the node communication radius r, that is, the one-hop distance, to ensure that data is not lost; the path from any node to the root node is not unique, but the number of hops through these paths is the same; that is, the depth of the node is unique to ensure that the path from any node to the root node is the shortest path, which is conductive to reduce the time delay of the aggregation process.

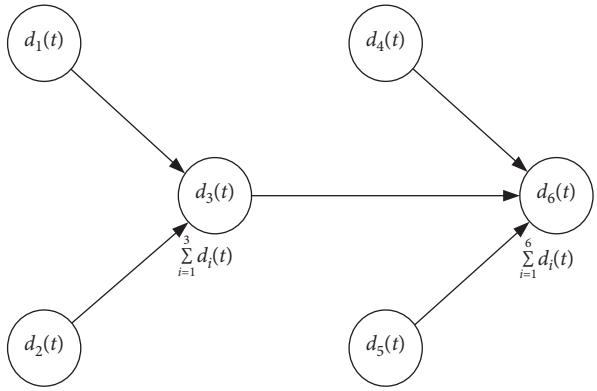


Figure 1: Schematic diagram of aggregation of SUM function.

At this phase, the specific process of constructing the fat tree on the randomly generated node set (the point set of wireless sensor network G) is as follows: firstly, the center point of G is selected as the Sink node S, which is the root node of the fat tree; then, the Sink node is used as the base node, and the node communication radius r is used as a hop distance to search for its child nodes, and the child nodes obtained are linked to the fat tree; then, the child nodes obtained are applied as the base node, and the node communication radius r is applied as the one-hop distance to search for its child nodes (the nodes that are not in the fat tree), and the process is looped until all the nodes are connected to the fat tree. In this way, the fat tree is successfully constructed, and the process of searching for the child nodes is shown in Figure 2.

In Figure 2, the child nodes searched by the Sink node S employing communication radius r are nodes 1, 2, and 3; the child nodes searched by node 1 employing communication radius r are nodes 4 and 5; the child nodes searched by node 2 employing communication radius r are nodes 5 and 6; the child nodes searched by node 4 employing the communication radius r are nodes 7 and 8, and so forth. Then, child nodes 1, 2, and 3 are linked to node S, nodes 4 and 5 to node 1, and nodes 5 and 6 to node 2, until all the nodes are linked, and the fat tree is constructed successfully. The fat tree structure is presented in Figure 3.

3.3. Slicing Phase. All the nodes in the network G cut their own sensory data into $n+1$ slices according to the number n of their parent nodes in the fat tree (the Sink node is not considered); retain 1 slice of them randomly, encrypt the rest of the data slices, and send to its parent nodes randomly. The schematic diagram is shown in Figure 4.

In Figure 4, node 5 is taken as an example to illustrate the data slices. Node 5 has three parent nodes 1, 2, and 3 in the fat tree. Therefore, the sensory data of node 5 is cut into 4 slices, namely, r_{51}, r_{52}, r_{53}, and r_{54}.

In that case, all the nodes in the network G have performed the sensory data slicing, which greatly improves the performance of the data privacy protection; in addition, the number of the parent nodes in the fat tree is known, which prevents the blind slicing and reduces the failure rate of slicing the data communication.

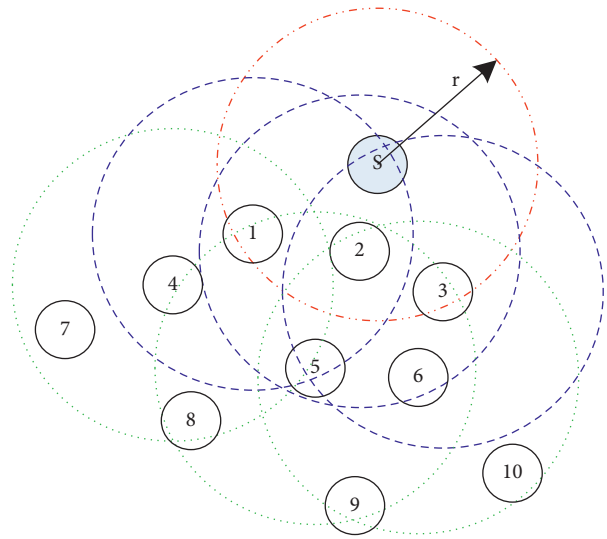


Figure 2: The process of searching for child nodes.

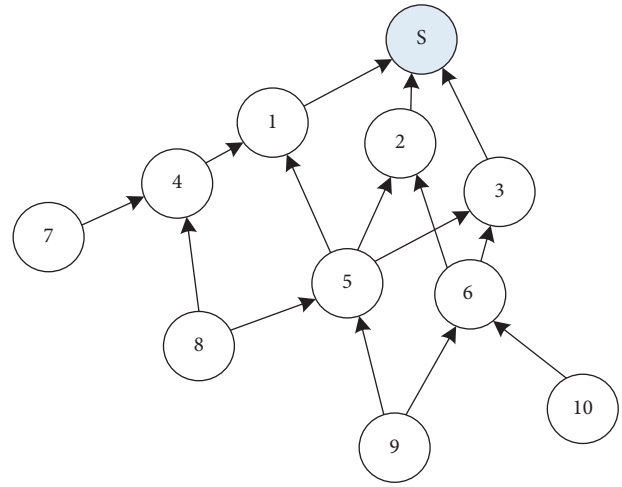


Figure 3: Fat tree structure.

3.4. Mixing Phase. After all the nodes in the network G receive the encrypted data slices, the shared key is used to perform decryption, and then a mixing calculation is performed on these data slices as well as the retained data slices. The mixed calculation process is illustrated in Figure 5. The SUM function is applied to carry out the data slice mixing calculation, as shown in

$$A_j = \sum_{i,j \in U_j} r_{ij}. \tag{4}$$

A_j is the new data packet after the mixing calculation in node j. U_j is the collection of node j and its child nodes i in the fat tree, as well as the collection of node j and node i that sends the data slice to node j. r_{ij} means the data packet of the data slice from node i to node j.

In Figure 5, node 5 is taken as an example to describe data mixing. Node 5 has three data slices before mixing, namely, slice r_{55} that is reserved by itself, data slice r_{85} received from node 8, and data slice r_{95} received from node 9.

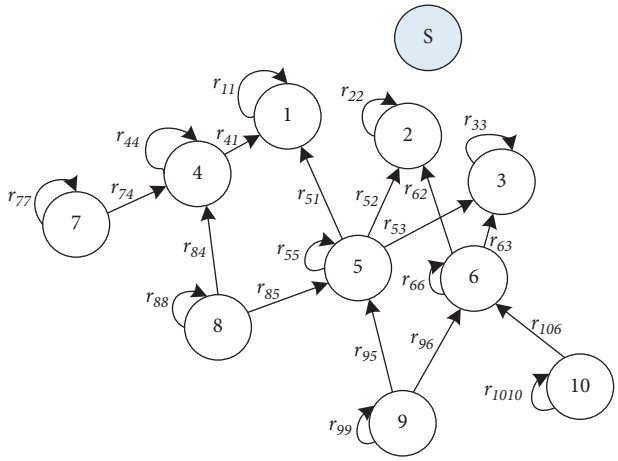


Figure 4: Schematic diagram of data slices.

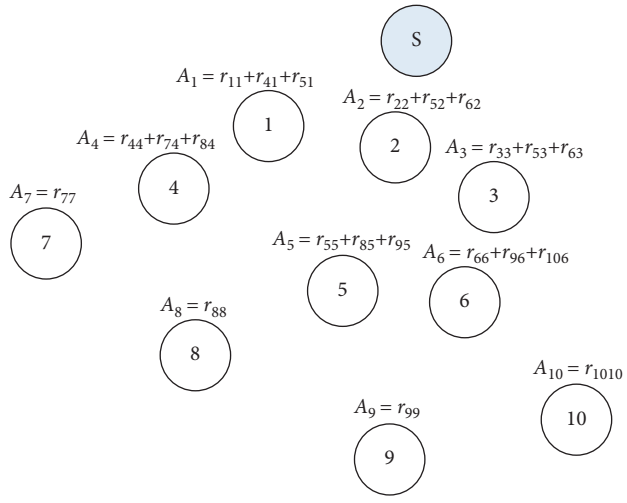


Figure 5: Schematic diagram of data mixing.

Node 5 performs the mixing calculation according to the SUM function to obtain the new data A_5 after mixing.

3.5. Aggregating Phase. A shortest path tree in the fat tree is selected as the aggregation tree, and after the mixing calculation, the new data packet is transferred up from all the leaf nodes to the Sink node layer by layer. In this process, after each node receives all the decrypted data, it performs a data aggregation calculation, and then the data encryption continues to be transmitted to the upper layer. According to the selected aggregation tree structure, the fixed time interval (i.e., time slice, the time to complete one-hop data transmission) is allocated to the node to ensure the completion of the data transmission. The aggregation scheduling process is shown in Figure 6.

The network starts the data transmission at time $t0$. After t time intervals, all the information is transmitted to the Sink node, and the aggregation cycle ends. Then, the aggregation cycle of the network is t (the data transmission time is much longer than the aggregation calculation time, and the set time interval is slightly longer than the data transmission time). As in Figure 6, the aggregation cycle of network G is 4.

Therefore, the aggregation cycle of the network G is fixed, which ensures the completion of data aggregation transmission, prevents the data package loss, and improves the aggregation accuracy.

4. Experimental Results and Analysis

The simulator embedded in TinyOS is used as a simulation tool to conduct simulation experiments on the privacy protection performance of the SMART algorithm [2], the D-SMART algorithm [12], and the proposed FTSMART algorithm. The simulation experiments incorporate the comparison experiment of data privacy protection performance, the comparison experiment of communication overhead, and the comparison experiment of aggregation accuracy.

4.1. Simulation Environment. In the simulation environment, the wireless sensor network is deployed as follows: 500 sensor nodes are randomly distributed in a two-dimensional rectangular area of 500 m ∗ 500 m. The specific parameter settings are shown in Table 1.

4.2. Privacy Protection Performance. After the nodes in the network use the slicing technology, if an attacker attempts to attain the data of a certain node, all the access links of this node must be cracked to restore its original data. *P(q)* is defined as the probability that the private data of a node is decrypted, and *P(q)* is used as a measure of the privacy protection, where q represents the probability that the link among the nodes is decrypted.

For the SMART algorithm, the probability of cracking the node privacy data is shown in

$$P(q) = q^{J-1} \sum_{k=0}^{d_{in\_\max}} P(\text{in} - \text{degree} = k) q^k, \tag{5}$$

where J expresses the number of the node data slice, q^{J-1} expresses the probability that the out-degree link of the node is cracked, k expresses the number of the in-degree links, $d_{in\_\max}$ expresses the maximum number of the in-degree links of the node and is determined by the number of the neighboring nodes, $P(\text{in-degree} = k)$ expresses the probability that the in-degree link is equal to k, and q^k expresses the probability that the in-degree link of the node is cracked. Therefore, the SMART algorithm is obviously affected by the number J of the node data slices.

For the D-SMART algorithm, the probability of cracking the node privacy data is shown in

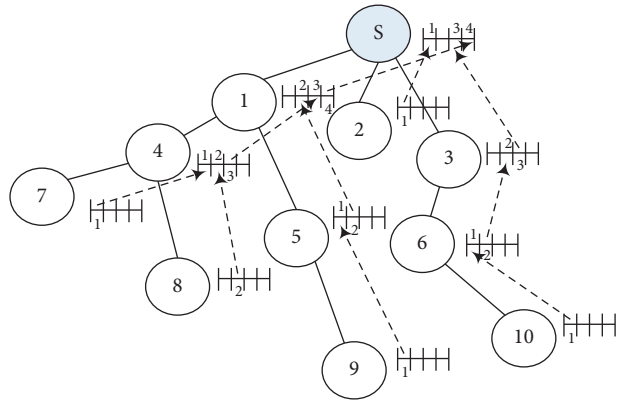


Figure 6: Schematic diagram of data aggregation scheduling.

Table 1: Simulation parameter settings.

Parameter	Implication	Value
N	Number of nodes	500
L	Simulation area (m^2)	500 ∗ 500
r	Communication radius (m)	25
Pa	Proportion of fusion nodes	20%

$$P(q) = \sum_{j\in\{1,2,3,4\}} P(\text{out} - \text{degree} = j)q^{j-1} * \sum_{k=0}^{d_{in\_\max}} P(\text{in} - \text{degree} = k)q^{k}, \tag{6}$$

where j represents the number of the out-degree links. For the leaf nodes, $j \in \{2, 3, 4\}$. For other nodes, $j = 1$, P(out-degree $= j$) represents the probability that the in-degree link is equal to j, q^{j-1} indicates the probability that the out-degree link of the node is cracked. Therefore, the D-SMART algorithm is obviously affected by the aggregation node ratio Pa.

For the FTSMART algorithm, the probability of cracking the node privacy data is shown in the following formula:

$$P(q) = \sum_{j=1}^{d_{out\_\max}} P(\text{out} - \text{degree} = j)q^{j-1} * \sum_{k=0}^{d_{in\_\max}} P(\text{in} - \text{degree} = k)q^{k}, \tag{7}$$

where $d_{out\_\max}$ signifies the maximum number of the out-degree links of the node and is determined by the number of the parent nodes in the fat tree, the direct child nodes j of the Sink node is equal to 1, $d_{in\_\max}$ is determined by the number of the direct child nodes in the fat tree, and only the leaf node k is equal to 0. Therefore, the FTSMART algorithm is affected by the fat tree structure to some extent.

The simulation experiment results of the privacy protection performance of the SMART algorithm, the D-SMART algorithm, and the proposed FTSMART algorithm are demonstrated in Figure 7.

It can be observed from Figure 7 that as the probability of cracking the communication link increases, the probability of the privacy data exposure of the SMART algorithm, the D-SMART algorithm, and the proposed FTSMART algorithm also increases monotonically. Among them, the FTSMART algorithm has the lowest probability of the privacy data exposure. The D-SMART algorithm only has the leaf nodes for the dynamic slice. As the proportion of the aggregation nodes Pa decreases, the privacy protection capability of the D-SMART algorithm exceeds that of the SMART algorithm. The FTSMART algorithm, like the SMART algorithm, slices all the node data. The SMART algorithm slices each node data and satisfies $J = 3$, and as J increases, the privacy protection capability is enhanced. The number of the data slices for each node of the FTSMART algorithm is determined by $n + 1$ slices according to the number n of their parent nodes in the fat tree. The distribution and density of the nodes in this experimental environments determine that the FTSMART algorithm has the highest privacy protection capability than the SMART algorithm and the D-SMART algorithm.

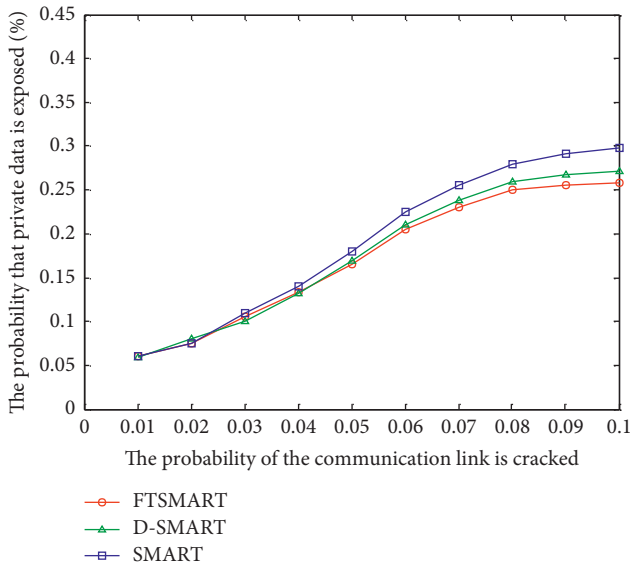


Figure 7: Comparison of data privacy protection performance.

4.3. Data Communication Overhead. In the wireless sensor network, the node energy consumption mainly derives from the data transmission among nodes. A large amount of the data transmission result in the premature death of some nodes, thus shortening the entire life cycle of the network. We adopt the total amount of the data packets transmitted in the network as a measure of the communication overhead. The simulation experiments are carried out to compare the communication overhead of the SMART algorithm, the D-SMART algorithm, and the FTSMART algorithm.

For the SMART algorithm, in the slicing phase, each sensor node cuts the sensory data into J slices and sends (J-1) slices to the neighboring nodes to generate (J-1) data packets. In the aggregating phase, each sensor node sends a new data packet after mixing to the upper node. Therefore, the communication overhead of the SMART algorithm is expressed in

$$O_c = N * (J + 1) + N * 1 = N * J, \tag{8}$$

where O_c signifies the communication overhead and N signifies the total number of the nodes in the network.

For the D-SMART algorithm, in the slicing phase, the aggregation node data does not need to be sliced. The leaf node data is dynamically sliced. In the aggregating phase, each sensor node needs to send one data packet to the upper node. Therefore, the communication overhead of the D-SMART algorithm is expressed in

$$O_c = N * Pa + \sum_{i=1}^{N*(1-Pa)} j_i \left(j_i \in \{2, 3, 4\}\right), \tag{9}$$

where (1-Pa) represents the proportion of the leaf nodes and j_i represents the number of the data slices of node i, that is, the total number of the data packets generated by node i.

For the FTSMART algorithm, in the slicing phase, all the sensor nodes need to slice the data into $(n+1)$ slices according to the number n of their parent nodes. In the aggregation phase, each sensor node needs to send one data packet to the upper node. Therefore, the communication overhead of the D-SMART algorithm is expressed in

$$O_c = \sum_{i=1}^{N} j_i \left(j_i \in [1, 2, \cdots, n_{\max} + 1]\right), \tag{10}$$

where i represents the node sequence number and $n_{\max}$ represents the maximum number of the parent nodes for all the sensor nodes.

The simulation experiment results of the communication overhead of the SMART algorithm, the D-SMART algorithm, and the proposed FTSMART algorithm are demonstrated in Figure 8.

It has been observed from Figure 8 that the network communication overhead of the SMART algorithm in different aggregation cycles is the highest, maintaining 1500 data packets. The network communication overhead of the D-SMART algorithm varies from 1100 to 1300 data packets. And the network communication overhead of the FTSMART algorithm is the lowest, maintaining 1100 data packets. In the SMART algorithm, the number of the node slices is fixed ($J = 3$), and the communication overhead is also fixed. In the D-SMART algorithm, the aggregation node ($Pa = 0.2$) does not participate in the slicing behavior. The leaf nodes change in the interval [2, 4] according to the number of different slices of the sensory data, and the communication overhead fluctuates as well. All the nodes of the FTSMART algorithm participate in the slicing behavior, and the number of the slices depends on the number of the parent nodes in the fat tree. The fat tree structure in the network is fixed, and the communication overhead is fixed accordingly.

4.4. Data Aggregation Accuracy. In wireless sensor network, the data aggregation accuracy is one of the important metrics that demonstrate the performance of the data aggregation algorithms. In theory, the accuracy of the aggregation result is 100%, which signifies that the ultimate aggregation result is equal to the sum of the data collected by each node in the network [6]. However, in practical applications, the data transmission collision, the delay, the bit error, and the packet loss in the data aggregation process are unavoidable, which mainly resulted from the use of shared channels to transmit data in wireless sensor network, thus making the accuracy of the aggregation result lower. The calculation of the data aggregation accuracy is illustrated in

$$P = \frac{D^*}{\sum_{i=1}^{N} D_i}, \tag{11}$$

where P denotes the data aggregation accuracy and D^* expresses the final aggregation result obtained by the Sink node. D_i represents the sensory data of node i, and $\sum_{i=1}^{N} D_i$ represents the sum of sensory data of all the nodes in the network.

The simulation experiment results of the aggregation accuracy of the SMART algorithm, the D-SMART algorithm, and the proposed FTSMART algorithm are demonstrated in Figure 9.

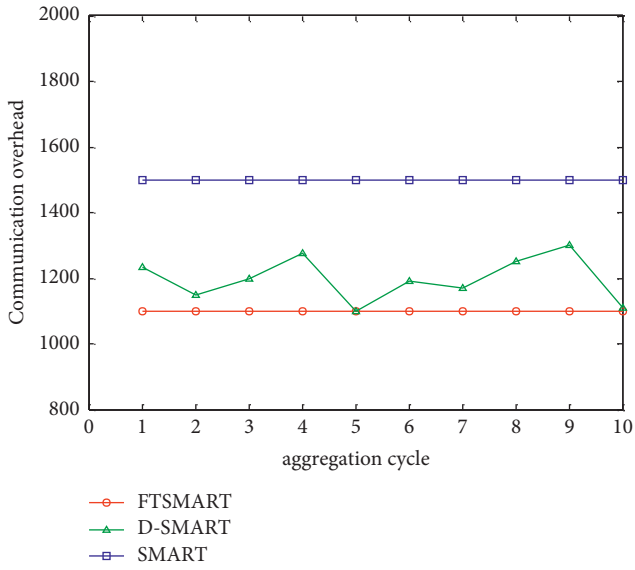


Figure 8: Comparison of communication overhead.

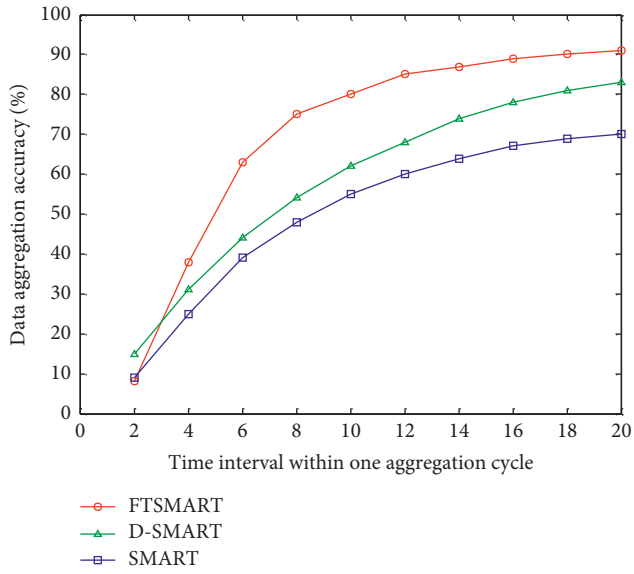


Figure 9: Comparison of aggregation accuracy.

It can be detected from Figure 9 that, within an aggregation cycle, as the time interval increases, the aggregation accuracy of these three comparison algorithms increases monotonically and the increase in speed becomes slower. The aggregation accuracy rates of the SMART algorithm, the D-SMART algorithm, and the proposed FTSMART algorithm are 70%, 83%, and 91%, respectively. Among them, the SMART algorithm has the lowest aggregation accuracy, and the FTSMART algorithm has the highest aggregation accuracy. Within the aggregation cycle, the more the data transmitted, the more the probability of collision in the transmission process, the more the data lost, and the greater the impact on the aggregation result. That is, an aggregation scheme with low communication overhead can obtain the favorable data aggregation accuracy. In the FTSMART algorithm, the aggregation accuracy is improved by optimizing the aggregation tree and reasonably allocating the time intervals.

5. Conclusion

On the basis of analyzing and researching the SMART algorithm and the relative algorithms, the FTSMART algorithm is proposed, and the fat tree is introduced into the data aggregation of wireless sensor network, which has greatly improved the deficiencies of the SMART algorithm in the data privacy protection and the aggregation accuracy. From the simulation experiments, it has been detected that the FTSMART algorithm is affected by the fat tree structure to some extent, and therefore the process of optimizing the fat tree to generate the aggregation tree can be taken into consideration. The research of the paper has set the conditions of the equal time interval transmission and the complete aggregation, which requires further expansion. In the future, the researches on the multimodal data aggregation methods and the multimodal data security protection will be the huge challenges.

Data Availability

All of the data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. Ramesh and P. K. Varshney, "Data aggregation techniques in sensor networks: a survey," *Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 48–63, 2006.

[2] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "PDA: privacy-preserving data aggregation in wireless sensor networks," in *Proceeding of the 26th IEEE International Conference on Computer Communications*, pp. 2045–2053, IEEE Computer Society Press, Washington, DC, USA, 2007.

[3] C. E. Leiserson, "Fat-trees: universal networks for hardware-efficient supercomputing," *IEEE Transactions on Computers*, vol. C-34, no. 10, pp. 892–901, 2012.

[4] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a Tiny AGgregation service for ad-hoc sensornet works," *Acm Sigops Operating Systems Review*, vol. 36, no. SI, pp. 131–146, 2002.

[5] H. Li, K. Lin, and K. Li, "Energy-efficient and high-accuracy secure data aggregation in wireless sensor networks," *Computer Communications*, vol. 34, no. 4, pp. 591–597, 2011.

[6] C. Li and Y. Liu, "ESMART: energy-efficient slice-mix-aggregate for wireless sensor network," *International Journal of Distributed Sensor Networks*, vol. 9, no. 12, pp. 1–9, Article ID 134509, 2013.

[7] C.-X. Liu, Y. Liu, Z.-J. Zhang, and Z.-Y. Cheng, "High energy-efficient and privacy-preserving secure data aggregation for wireless sensor networks," *International Journal of Communication Systems*, vol. 26, no. 3, pp. 380–394, 2013.

[8] Q. Liang and Y. A. N. G. Geng, "A low traffic privacy-preserving aggregation algorithm," *Computer Technology and Development*, vol. 23, no. 8, pp. 133–136, 2013.

[9] T. C. Wang, X. L. Qin, L. Liu et al., "Secure and energy-efficient spatial data aggregation algorithm in wireless sensor networks," *Journal of Software*, vol. 25, no. 8, pp. 1671–1684, 2014.

[10] G. Yang, A.-Q. Wang, Z.-Y. Chen, J. Xu, and H.-Y. Wang, "An energy-saving privacy-preserving data aggregation algorithm," *Chinese Journal of Computers*, vol. 34, no. 5, pp. 792–800, 2011.

[11] L. I. Sen and Y. A. N. G. Geng, "Research on precision aggregation privacy-preserving algorithm in wireless sensor networks," *Computer Technology and Development*, vol. 23, no. 9, pp. 139–142, 2013.

[12] J. Wang and Y. Chen, "Research and improvement of wireless sensor network secure data aggregation protocol based on SMART," *International Journal of Wireless Information Networks*, vol. 25, no. 3, pp. 232–240, 2018.

[13] G. Yang, S. Li, Z. Y. Chen et al., "High-accuracy and privacy-preserving oriented data aggregation algorithm in sensor networks," *Chinese Journal of Computers*, vol. 36, no. 1, pp. 189–200, 2013.

[14] J. Shi, R. Zhang, Y. Liu et al., "PriSense: privacy-preserving data aggregation in people-centric urban sensing systems," in *2010 Proceedings IEEE INFOCOM*, pp. 758–766, San Diego, CA, USA, March 2010.

[15] W. He, H. Nguyen, X. Liu et al., "iPDA: an integrity-protecting private data aggregation scheme for wireless sensor networks," in *Proceeding of MILCOM 2008 - 2008 IEEE Military Communications Conference*, pp. 1–7, San Diego, California, USA, November 2008.

[16] V. Kumar and S. Madria, "PIP: privacy and integrity preserving data aggregation in wireless sensor networks," in *Proceeding of 2013 IEEE 32nd International Symposium on Reliable Distributed Systems*, pp. 10–19, IEEE, Braga, Portugal, September 2013.

[17] K. Parmar and D. C. Jinwala, "Malleability resilient concealed data aggregation in wireless sensor networks," *Wireless Personal Communications*, vol. 82, no. 1, pp. 777–798, 2015.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5599962, 8 pages
https://doi.org/10.1155/2021/5599962

Research Article

Information Security Field Event Detection Technology Based on SAtt-LSTM

Wentao Yu, Xiaohui Huang, Qingjun Yuan, Mianzhu Yi, Sen An, and Xiang Li

PLA Strategic Support Force Information Engineering University, Zhengzhou 450000, China

Correspondence should be addressed to Mianzhu Yi; 381746262@qq.com

Received 19 January 2021; Revised 26 May 2021; Accepted 21 June 2021; Published 15 August 2021

Academic Editor: Liguo Zhang

Detecting information security events from multimodal data can help analyze the evolution of events in the security field. The Tree-LSTM network that introduces the self-attention mechanism was used to construct the sentence-vectorized representation model (SAtt-LSTM: Tree-LSTM with self-attention) and then classify the candidate event sentences through the representation results of the SAtt-LSTM model to obtain the event of the candidate event sentence types. Event detection using sentence classification methods can solve the problem of error cascade based on pipeline methods, and the problem of CNN or RNN cannot make full use of the syntactic information of candidate event sentences in methods based on joint learning. The paper treats the event detection task as a sentence classification task. In order to verify the effectiveness and superiority of the method in this paper, the DuEE data set was used for experimental verification. Experimental results show that this model has better performance than methods that use chain structure LSTM, CNN, or only Tree-LSTM.

1. Introduction

Research fields such as intelligence analysis, behavior modeling, and computational attribution models in the security field urgently need the support of a large amount of behavioral knowledge from the real world. Therefore, extracting information security behavior knowledge from massive open-source texts has become one of the current core research topics. Facing the security field, the thesis systematically studies the detection methods of behavioral knowledge and event information in massive open-source texts. By analyzing the evolution of events in the security field, the development prediction of information security events and the effectiveness evaluation of intervention actions can be realized. As an important branch of the natural language processing field, the purpose of the information extraction technology is to help users to complete information judging, comparing, and retrieving quickly and accurately. Meanwhile, as one of the key subtasks of information extraction technology, event extraction technology is becoming an important and challenging research hotspot in the field of natural language processing.

Around information extraction technology, a series of international evaluation conferences have been held, such as MUC (Message Understanding Conference), ACE (Automatic Content Extraction), and DUC (Document Understanding Conference). These conferences provide a standard test platform for the researchers who study information extraction technology and effectively improve the research level of information extraction technology. In a word, these conferences actively promoted the development of information extraction technology [1]. The event templates defined by ACE in 2005 have become a defining standard of event types generally recognized by the majority of event extraction technology researchers; these templates divide the events into 8 main types and 33 subtypes and provide corresponding training and testing corpus based on relevant templates.

There are two tasks among event extraction processes: event detection and extraction of the role of event argument. Generally speaking, almost all texts are unlabeled, such as news information in newspapers. Though these texts carry a lot of information, not every text expresses a separate event. Moreover, different event types involve different roles of

argument. To accurately and clearly extract the relevant information about the event from complex and diverse texts, event detection should be the primary task to be completed. The task of event detection is to distinguish whether a statement contains an event and which type of event it contains. The detection accuracy has a crucial impact on the subsequent identification of the role of argument. Meanwhile, the inaccurate event type is an important reason for the incorrect cascade of event extraction tasks. Currently, event trigger words have a significant impact on event detection. The definition of an event trigger is a word or phrase that can clearly express the occurrence of an event. However, the trigger word is not an indispensable basis for event detection. This paper focuses on the research of event detection tasks, with the purpose of improving the detection rate. This study has important significance for improving the event extraction technology. An efficient event extraction scheme can accurately extract events and event elements in the information security field, organize related social behaviors and behavioral premises and result knowledge, and construct an information security affair graph, which can be used in application scenarios such as causal reasoning in the information security field.

In view of these problems above, we propose a Tree-LSTM network with self-attention mechanism for the event detection model. The main research content of this paper includes the following aspects:

(1) In order to solve the restriction of the trigger word recognition performance, we treat the event detection task as a sentence classification task for processing.

(2) In the text preprocessing stage, we use existing natural language processing tools to perform word segmentation and dependency syntax analysis on text and use the GloVe word vector dictionary based on Wikipedia Chinese prediction training to vectorize the segmented data set.

(3) In order to update the hidden state results obtained by the LSTM network, we introduce a self-attention mechanism in the network training.

(4) In terms of network, we use the dependency tree structure of sentences parsed by NLP tools as the input of the tree structure LSTM network. Tree-LSTM overcomes the limitation that the linear chain structure of LSTM only allows strict sequence tasks and combines the syntactic information of the text.

The method in this paper uses syntactic information to recognize event types through event sentence classification. In the next step, researchers can use event types to assist in the joint extraction of trigger words and arguments, which not only effectively utilizes the dependence between arguments and trigger words but also can avoid error cascade.

2. Background

From the perspective of the development process, there are three main categories of event extraction technologies: method based on pattern matching, machine learning, and neural network [2].

2.1. Method Based on Feature Engineering. Before 2015, the working idea of event extraction is mainly focused on pattern matching or artificially designed features. The most prominent feature of the method based on pattern matching is that it could have good results in some specific fields, but its portability is so poor [1, 3].

2.2. Method Based on Traditional Machine Learning. The method based on traditional machine learning largely solves the poor portability of the pattern matching-based method but it is highly dependent on the labeled data. For example, Ahn divided the event detection task evaluated by ACE into 4 steps: identify event trigger words, tag event role, tag attributes, and detection output [4]. The first two steps complete the event extraction, and the rest of the steps complete the event detection task. Although these methods are more modular, easier to implement based on common components, and faster to run and debug, like other technologies based on trigger words, these methods do not pay enough attention to the multimeaning phenomenon of the text [5]. Moreover, the event detection task in these methods greatly affects the subsequent event element identification and classification effect of the event extraction system due to the error cascade problem that occurs in each step (the extraction of arguments in the pipeline method depends on the recognition result of the trigger word. If the trigger word is recognized incorrectly, it will seriously affect the recognition of the argument). Furthermore, it is time-consuming to select trigger words from sentences for labeling, and the cost of annotating the training corpus is also relatively expensive. To sum up, the recognition performance of trigger words has become a bottleneck restricting the efficiency of event detection tasks.

2.3. Method Based on Neural Network. Since 2015, some researchers have tried to use CNN/RNN for event extraction [6]. Chen et al. proposed an event extraction model called Dynamic Multipool Convolutional Neural Network (DMCNN). This model encodes each word in the sentence into a word embedding vector and adds the relative position embedding vector as a feature of auxiliary event type classification. Considering that a sentence may contain the emptying of multiple events, the model adopts a dynamic multipool approach [7]. Xu et al. proposed a Chinese event detection method based on multifeature fusion and Bi-LSTM, in which contextual information is captured based on the Bi-LSTM model. This method performs well on the CEC data set [8]. Zhang et al. proposed a new framework for event extraction based on an inverse reinforcement learning method using a generative adversarial network (GAN) and delved into the influence of two different pretraining methods of ELMo and word2vec on the proposed model [9]. Liu proposed a novel multilingual approach—dubbed as Gated Multilingual Attention (GMLATT) framework—to

simultaneously address data scarcity and monolingual ambiguity issues found in event detection [10]. To enhance both automatic feature selection and classification, Shen et al. presented an end-to-end convolutional highway neural network and extreme learning machine (CHNN-ELM) framework to detect biomedical event triggers [2]. Yu et al. built an event extraction model by using Tree-LSTM and Bi-GRU. The model obtains the vectorized representation results of candidate event sentences through Tree-LSTM and Bi-GRU, obtains the event type through sentence classification, and uses the event type as part of the event extraction input to further improve the performance and efficiency of the event extraction model [11].

3. Event Detection Model Based on SAtt-LSTM

To improve the performance of event detection, we designed an event detection model based on self-attention mechanism and Tree-LSTM. First, the model uses syntactic analysis tools to perform dependency syntactic analysis on candidate event sentences to obtain the dependency tree structure of the text. Then, we use the Chinese Wikipedia corpus to pretrain the GloVe word vector and get the word vector dictionary. Subsequently, the event sentence content document, vocabulary list, and word vector dictionary obtained after training are established for the DuEE data set to obtain the word vector representation of each word. Next, we introduce the obtained word vector into a self-attention mechanism to learn the semantic related features between discontinuous words in a sentence. The purpose of this step is to combine the obtained word vector with the semantically related features. Moreover, we combine the results above with the parsed dependency tree structure and input it into the Tree-LSTM neural network to learn sentence structure features and then obtain the vectorized representation of the candidate event sentence. Finally, we input the feature representation of the sentence into the Softmax classifier to get the final event detection result. Figure 1 is the model architecture diagram.

3.1. Text Preprocessing Module. The main purpose of dependency syntax analysis is to analyze the grammatical structure of a sentence and express it as an easy-to-understand structure. In this paper, as the input of the Tree-LSTM network, the event sentence should be constructed into a dependency tree structure firstly. In order to obtain suitable syntactic analysis tools for our model, we mainly choose two NLP tools, LTP and HanLP, to analyze the dependent syntax of sentences, respectively. Taking the sentence "Chinese public organizations have been attacked by foreign hackers using ransomware" as an example, Figure 2 shows the sentence dependency syntax tree.

Judging from the results of syntactic analysis of example sentences, the root node of the dependency tree is likely to be the trigger word in the event sentence. This proves that syntactic analysis could make good use of syntactic information, so as to carry out more accurate event element detection and event type discrimination.

When using LTP for dependency syntax analysis, we need to segment the sentence, mark the part of speech first, and then realize the dependency syntax analysis based on the results of the word segmentation and part of speech tagging. Then, according to the results of word segmentation and part-of-speech tagging, the dependency syntax analysis is realized; finally, the index and word segmentation results representing the parent node of the dependency arc are obtained as shown in Figure 3.

3.2. Sentence-Vectorized Representation Module. The dependency tree obtained after the dependency syntax analysis of the event sentence is used as the input of Tree-LSTM. Each word in the sentence is a node in the tree. Sentences are input to the Tree-LSTM network to obtain a hidden state value of each event sentence, the value is used as the output of the network, and the final judgment of the event type is based on the output [12, 13].

After getting the initial hidden state, call the self-attention mechanism model to update the hidden state and then combine the self-attention information to get the final hidden state result. The state is used for the final classification module [14].

In the process of network forward training, for each node k in the tree structure, c_k is the set of child nodes of node k and is the sum of the hidden state of the child nodes of k:

$$\widetilde{h}_k = \sum_{i \in c(j)} h_i. \tag{1}$$

The memory storage unit of Tree-LSTM is designed with a forget gate for each node. For node k, the input gate is transformed in the following way:

$$i_k = \sigma\left(W^{(i)} x_k + U^{(i)} \widetilde{h}_k + b^{(i)}\right). \tag{2}$$

The output gate is o_k switched in the following way:

$$o_k = \sigma\left(W^{(o)} x_k + U^{(o)} \widetilde{h}_k + b^{(o)}\right). \tag{3}$$

The forget door of child node i f_{ki} is switched in the following way:

$$f_{ki} = \sigma\left(W^{(f)} x_k + U^{(f)} \widetilde{h}_i + b^{(f)}\right). \tag{4}$$

Here, W and U are learnable parameters and b is the bias term.

Through this forget gate, the parent node could selectively forget the child node. Meanwhile, the input gate could be adjusted according to the semantic importance of the child node. For each node, the input gate collects all syntactic and semantic information from its child nodes, and the output gate balances meaningful information from its child nodes [15]. Finally, we designed a method to merge syntactic and semantic information into the gates of input, output and forget, and an explicit method that directly merges syntactic and semantic information into the hidden state of the node. The last step of the method is to implement a vectorized representation of the root of each tree.

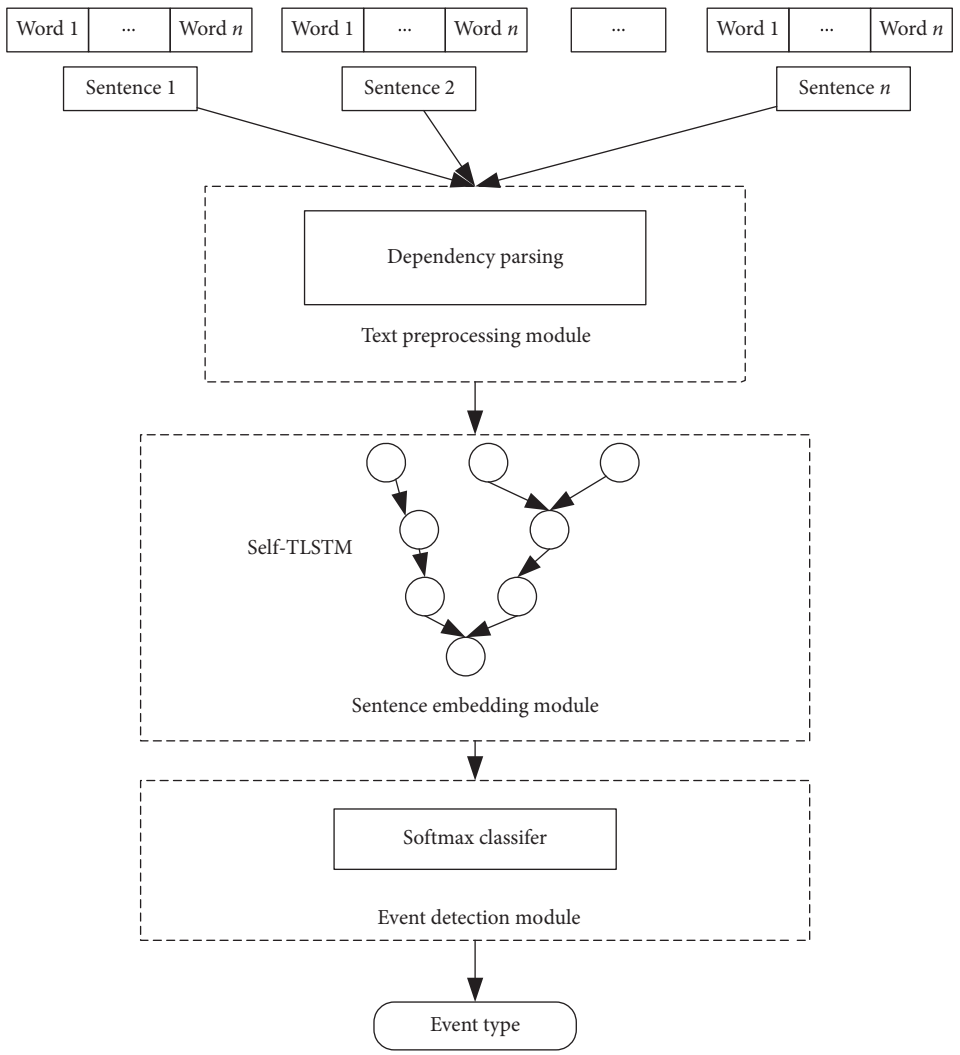


Figure 1: Model architecture.

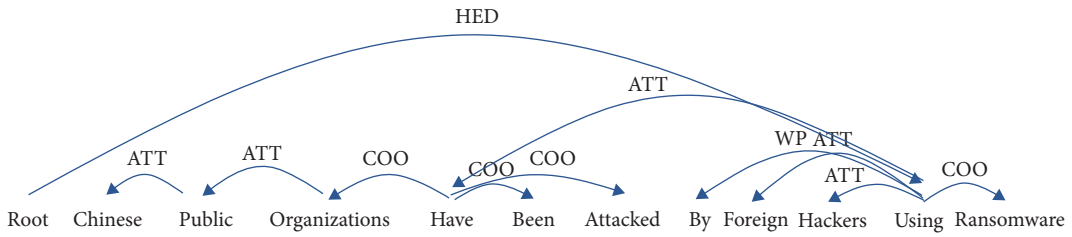


Figure 2: Display diagram of dependency syntax tree.

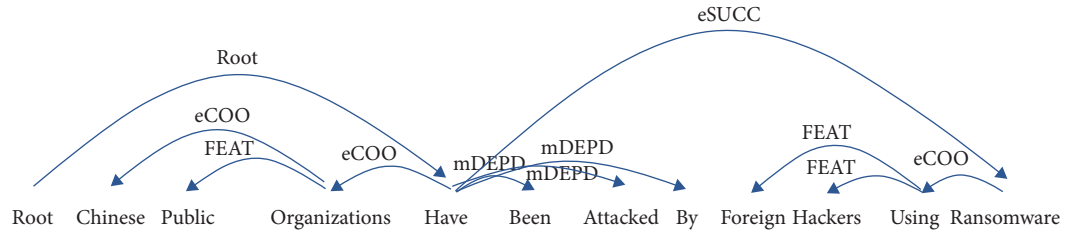


Figure 3: Example sentence processing results by LTP.

In this paper, we set the self-attention mechanism as a layer. After passing the sentence into the Tree-LSTM network, we get the initial hidden state result h_k and then the hidden state matrix: $H = [h_1, h_2, \ldots, h_n]$. In the matrix above, n is the length of a given sentence, that is, the number of words.

The idea of self-attention is to establish a connection between any two words in a sentence through a calculation step to obtain those long-distance-dependent features in the sentence. The formula is as follows:

$$g_i = \sum_{i \neq j} \alpha_{i,j} \cdot h_j. \tag{5}$$

In the formula, $\alpha_{i,j} > 0$ is the attention weight. And through Softmax regularization technology, the attention weight $\sum_j \alpha_{i,j} = 1$ is calculated as follows:

$$\alpha_{i,j} = \frac{e^{\text{score}(h_i, h_j)}}{\sum_j e^{\text{score}(h_i, h_j)}},$$
$$\text{score}(h_i, h_j) = V_\alpha^T \tan \text{h}\left(w_\alpha \left[h_i \oplus h_j\right]\right). \tag{6}$$

In the formula above, score(h_i, h_j) is achieved through MLP. MLP is used to model the correlation of words to the hidden state (h_i, h_j). The coding and decoding flow diagram of the self-attention layer is shown in Figure 4.

The self-attention mechanism is an improvement of the attention mechanism [16], which reduces the dependence on external information and is better at capturing the internal correlation of data or features. In this paper, the self-attention mechanism is used to weigh the contribution of words in determining the type of event, which effectively improves the efficiency of the model. We input the obtained hidden state matrix into the self-attention layer to update the hidden state result and use the result as the final event type classification. Figure 5 shows its structure.

3.3. Event Detection Module. In the paper, the event detection task is essentially a classification task. Its purpose is to classify each word in the sentence to see if there are elements necessary to form an event. If it is confirmed as an event sentence, then classify its event type. The classification described above is also a multiclass problem. The formula is as follows:

$$S_i = \frac{e^i}{\sum_j e^j}. \tag{7}$$

In the formula, e^i represents the i-th element in the j-dimensional array and S_i is the Softmax classification value. In multiclassification problems, the class label y could have more than two values. In the multiclassification process, Softmax maps the output of multiple neurons to the interval (0, 1). It could be understood as a probability. Based on this probability value, multiclassification is complete. The category with the largest probability in the j-dimensional array is the final classification result. It means that the element with the largest value is marked as the final classification result.

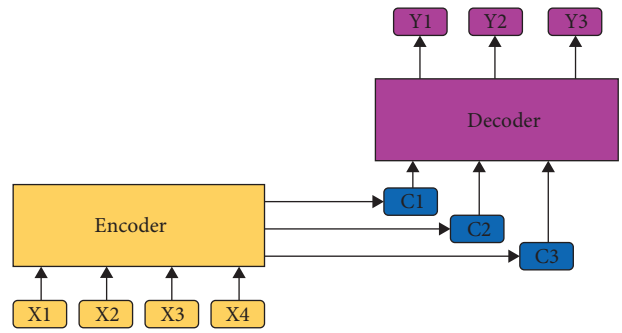


Figure 4: Structure of self-attention mechanism.

In the event detection process, after inputting the sentence vector to the Tree-LSTM network, a hidden state representation will eventually be obtained: h_k. For the hidden state of each node, Softmax classifier is used to classify the event type results. Comparing the predicted results with the actual real results, the minimized negative log-likelihood loss will be gotten. The formula is as follows:

$$\text{MLE} = \frac{1}{N} \sum_{n=1}^{n} \ln\left(1 + e^{-y_n w^T x_n}\right). \tag{8}$$

Here, x is the input value and y is the classification label. Then, the parameters of the model are optimized by the loss value.

4. Experimental Results and Analysis

4.1. Experimental Setup

4.1.1. Data Set. The data set used in the experiment is the DuEE data set provided by Baidu NLP [17]. It provides 11985 data sets that have been trained. These training data have identified the event type and marked the role of the argument. This data set divides the events into 9 types including 65 subtypes: life, judicial behavior, communication, product behavior, competition behavior, organizational relationship, finance/transaction, disaster/accident, and organizational behavior.

The experimental platform in the experiment is shown in Table 1.

4.1.2. Evaluation Index. Since the DuEE used in the paper is a Chinese data set, the pretraining of GloVe word vectors uses Wikipedia Chinese corpus. This corpus data covers a wide range and involves rich vocabulary. The paper borrows the trained Chinese Wiki Glove word vector dictionary [12]. The dictionary contains vector representations of 970,000 words, and the vector representation of each word is 100 dimensions.

4.1.3. Parameter Settings. All the experimental parameters in the experiment are shown in Table 2.

4.1.4. Evaluation Index. The paper uses precision (P), recall (R), and F1 value to evaluate the performance of the model.

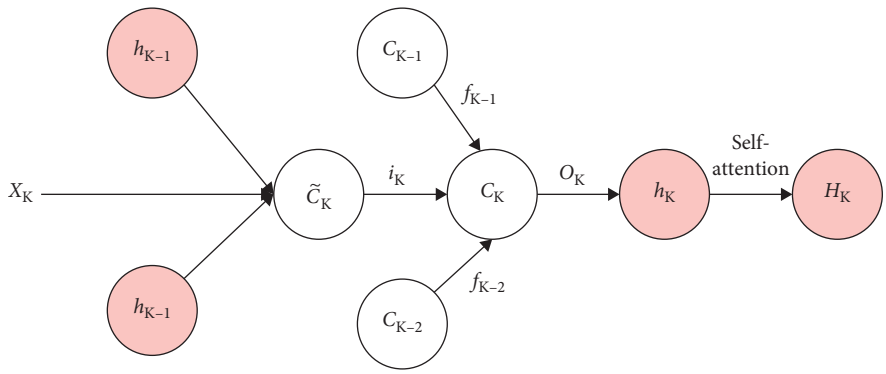


Figure 5: Structure of sentence embedding module.

Table 1: Experimental platform table.

Parameter	Value
OS	Win10
CPU	Intel Core i7-7700HQ
Memory size	16 G
Language	Python 3.6
Deep learning framework	PyTorch

Table 2: Model parameter table.

Parameter	Value
Learning rate	0.005
Batch size	25
Embedding learning rate	0.1
Weight decay	$1e-4$
Optimizer	Adagrad
Fine grain	2
Word embedding size	100
Attention dim	350

Among the formulas, TP is a positive example that is correctly identified, FP is a negative example that is classified as a positive example, and FN is a positive example that is classified as a negative example. The precision rate indicates the proportion of the samples that the model predicts to be positive, and the recall rate indicates the proportion of the positive examples that are correctly identified. The focus of the two is different, and simply considering one of them cannot reflect the performance of the model. Therefore, the paper adds the F1 value between the two as the evaluation standard. In addition, in order to evaluate the overall performance of the multiclass model, the paper adopts the overall evaluation method of macroaverage, to calculate the overall F1 value by calculating the TP, FN, and FP of all samples.

4.2. Results and Analysis. The paper selects the following methods as the baseline method to compare the performance of event detection: (1) event detection model based on pattern matching; (2) expanding the trigger vocabulary for event detection; (3) event detection model based on LSTM; (4) event detection model based on Bi-LSTM; (5) event detection model based on Tree-LSTM.

4.2.1. Experimental Results. It could be seen from Table 3 that the Tree-LSTM event detection model based on the self-attention mechanism has achieved excellent results in all aspects compared to other methods. In terms of accuracy index, the model in this paper is slightly lower than the event detection model using Bi-LSTM network. Compared with other models, it shows a significant improvement. Meanwhile, the recall rate and F1 have also been significantly improved.

4.2.2. Experiment Analysis. Analyzing the experimental results, we can see that this model is superior to other models for the following reasons:

(1) Model Selection. Compared with the previous RNN model, LSTM adds the concept of forgetting gate. Through the training process, the model could learn what information should be remembered and what information should be forgotten and could do well to capture longer-distance dependencies. Therefore, compared with previous methods based on pattern matching and expanded trigger vocabulary, LSTM has a significant improvement in effect and its portability is better. Then, the Bi-LSTM-based method is used to capture sentence dependencies in two directions through the network, which improves the understanding of text semantics. Therefore, compared with LSTM, the accuracy rate is increased by nearly 5 percentage points, and the F1 value is also increased by about 8 percentage points. Compared with Bi-LSTM, Tree-LSTM uses a tree topology, and its input of the network is the sentence processed by the NLP tool. Through this approach, it could better combine the syntactic information of sentences and avoid the problem that the dependency between long-distance word pairs is difficult to obtain. Compared with the Bi-LSTM-based method, the recall rate is increased by 6.4 percentage points. In this paper, we select a SAtt-TLSTM-based model. Based on Tree-LSTM, we introduce a method that could directly calculate the dependency relationship regardless of the distance between words to learn the internal structure of a sentence. To update the initial hidden state of the network output, we implement a relatively simple parallel computing self-attention mechanism. Compared with Tree-LSTM, the recall rate of this model has increased by nearly 7 percentage

Table 3: Baseline methods and results.

Baselines	P	R	F1
Pattern matching	0.627	0.508	0.561
Expanded trigger vocabulary	0.571	0.642	0.697
LSTM	0.676	0.655	0.626
Bi-LSTM	0.720	0.680	0.699
Tree-LSTM	0.634	0.744	0.709
SAtt-LSTM	0.691	0.812	0.731

points, and the F1 value has also increased by 2.2 percentage points. It should be the best model among all baseline methods.

(2) Application of Self-Attention Mechanism. The paper introduces a self-attention mechanism that could learn the correlation between the current word and the previous part of the sentence. It could judge the importance of the current word according to the strength of the correlation. To a certain extent, it solves the problem of the inability to highlight keywords. The reason for the problem is that the model in the paper does not remove the stop words. In other words, the critical words should be the trigger words in an event. After combining the self-attention mechanism with the original dependency tree result, the trigger word becomes more prominent. The above thesis has great advantages in the discrimination of event types. The results show that the model combined with the self-attention mechanism performs better than the Tree-LSTM model without the self-attention mechanism.

4.2.3. Model Performance Factors

(1) Word Segmentation Mechanism. The result of word segmentation is an important element that affects the experiment process and results. Through experimental comparison, we found that using different tools to do word segmentation and syntactic analysis of the same text finally got different indicators. As shown in Table 4 and Figure 6, the final result obtained by using LTP is slightly better than that of HanLP. However, in the training process, the text using LTP word segmentation and dependent syntax analysis requires longer time for training, and it also needs more rounds to reach the optimal parameters. In fact, whether it is LTP or HanLP used in the paper or other systems such as Stanford NLP or jieba word segmentation, its ability to recognize unregistered words needs to be improved, and an external dictionary is needed to expand its thesaurus. These defects double the workload of preprocessing. Furthermore, it is basically impossible to build a dictionary containing all knowledge bases. At present, the Chinese classification of "words" has not formed an officially recognized unified standard. This is also one of the key reasons for the difficulty of Chinese word segmentation. Different segmentation results are particularly important for accurately obtaining its word vector. For those words with special symbols, if the symbols and words cannot be separated well during word segmentation, it will lead to the inability to find the corresponding word vector in the word vector, which will affect the final experimental results.

Table 4: Word segmentation methods and results.

Word segmentation method	P	R	F1
LTP	0.698	0.812	0.734
HanLP	0.691	0.812	0.731

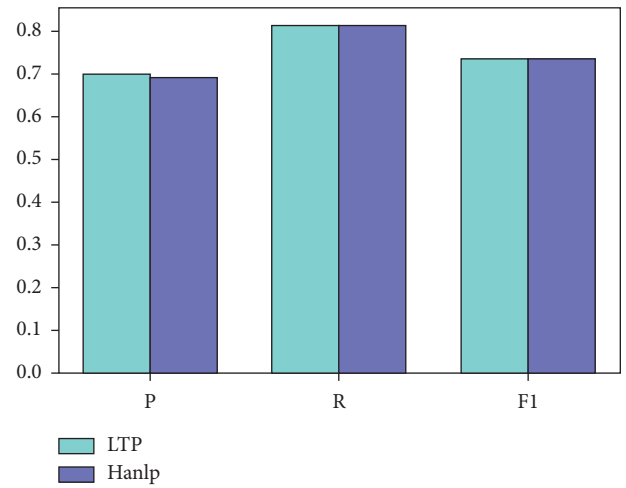


Figure 6: The effect of different word segmentation methods.

(2) Removing Stop Words. In this model, after the dependent syntactic analysis of the event sentence, the word segmentation result is directly obtained from the syntactic analysis result. Then, in order to integrate the embedded sentence vector and the size of the dependency tree, the model does not remove stop words from the word segmentation results. In the LSTM model, in order to test whether the removal of stop words has an effect on the final result, we conducted comparative experiments, respectively, and the results proved that better experimental results could be obtained by removing the stop words. Stop words are mostly auxiliary words of connecting sentences, which have little effect on the actual content of the text. Therefore, after removing the stop words in the sentence, the core elements of the sentence could be better analyzed, and then the better analysis results could be obtained.

5. Conclusions

The paper focuses on the critical first step in the event extraction task. The result of the event detection task could directly affect the subsequent extraction of the entire event. Aiming at the shortcomings of previous methods, we treat the event detection task as a sentence classification task. By using the Tree-LSTM network that introduces a self-attention mechanism, we propose an event detection model. Firstly, the model uses the GloVe word vector to express the text vocabulary, starting from the bottom, focusing on the use of global information in the text. Next, the dependency tree of candidate event sentences is used as the input of the model to solve the shortcomings of insufficient use of

syntactic information in existing event detection models. Then, we use the Tree-LSTM network which introduces the self-attention mechanism to obtain the vectorized representation of the candidate event sentence. Finally, the task of event detection was completed by using Softmax. After training and testing on the DuEE dataset, the results obtained prove that the model proposed in the paper is superior to the previous model. It not only improves the accuracy of event detection but also provides good basic information for the role classification task of event extraction. This model fundamentally optimizes the problem of error cascade caused by incorrect trigger word recognition in the task of event extraction. At the same time, it further proves the feasibility and advantages of deep learning in event detection and event extraction technology.

Data Availability

The Baidu previously reported DuEE1.0 data were used to support this study and are available at https://aistudio.baidu.com/aistudio/competition/detail/65. The datasets are cited at relevant places within the text as references.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 300–309, HLT, San Diego, CA, USA, June 2016.

[2] C. Shen, H. Lin, X. Fan et al., "Biomedical event trigger detection with convolutional highway neural network and extreme learning machine," *Applied Soft Computing*, vol. 84, 2019.

[3] B. L. Hu, R. F. He, H. Sun, and W. J. Wang, "Chinese event type recognition based on conditional random fields," *Progress in Artificial Intelligence*, vol. 25, no. 3, pp. 445–449, 2012.

[4] D. Ahn, "The stages of event extraction," in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pp. 1–8, Sydney, Australia, July 2006.

[5] K. Zhang, Y. Guo, X. Wang, J. Yuan, Z. Ma, and Z. Zhao, "Channel-wise and feature-points reweights DenseNet for image classification," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, September 2019.

[6] X. Fan, H. Lin, Y. Diao, and Y. Zou, "An integrated biomedical event trigger identification approach with a neural network and weighted extreme learning machine," *IEEE Access*, vol. 7, pp. 83713–83720, 2019.

[7] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, vol. 1, pp. 167–176, Beijing, China, July 2015.

[8] G. Xu, Y. Meng, X. Zhou, Z. Yu, X. Wu, and L. Zhang, "Chinese event detection based on multi-feature fusion and BiLSTM," *IEEE Access*, vol. 7, pp. 134992–135004, 2019.

[9] T. Zhang, H. Ji, and A. Sil, "Joint entity and event extraction with generative adversarial imitation learning," *Data Intelligence*, vol. 1, no. 2, pp. 99–120, 2019.

[10] J. Liu, Y. Chen, K. Liu, and J. Zhao, "Event detection via gated multilingual attention mechanism," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[11] W. Yu, M. Yi, X. Huang, X. Yi, and Q. Yuan, "Make it directly: event extraction based on tree-LSTM and Bi-gru," *IEEE Access*, vol. 8, pp. 14344–14354, 2020.

[12] W. Liu, P. Liu, Y. Yang, Y. Gao, and J. Yi, "An attention-based syntax-tree and tree-lstm model for sentence summarization," *International Journal of Performability Engineering*, vol. 13, no. 5, p. 775, 2017.

[13] M. Ahmed, M. Rifayat Samee, and R. E. Mercer, "Improving tree-LSTM with tree attention," 2019, https://arxiv.org/abs/1901.00066.

[14] L. Mou, G. Li, L. Zhang, T. Wang, and Z. Jin, "Convolutional neural networks over tree structures for programming language processing," 2016, https://arxiv.org/abs/1409.5718.

[15] S. MacAvaney, L. Soldaini, A. Cohan, and N. Goharian, "GU IRLAB at SemEval-2018 task 7: tree-LSTMs for scientific relation classification," 2018, https://arxiv.org/abs/1804.05408.

[16] K. Zhang, N. Liu, X. Yuan et al., "Fine-grained age estimation in the wild with attention LSTM networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 9, pp. 3140–3152, 2020.

[17] X. Li, F. Li, L. Pan et al., "DuEE: a large-scale dataset for Chinese event extraction in real-world scenarios," in *Proceedings of the International Conference on Natural Language Processing and Chinese Computing*, pp. 534–545, Springer, Zhengzhou, China, October 2020.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9968905, 14 pages
https://doi.org/10.1155/2021/9968905

Research Article

Differentially Private Web Browsing Trajectory over Infinite Streams

Xiang Liu,[1] **Yuchun Guo**,[1] **Xiaoying Tan**,[2] **and Yishuai Chen**[1]

[1]*Beijing Jiaotong University, Beijing, China*
[2]*China Justice Big Data Institute Co., Ltd., Beijing, China*

Correspondence should be addressed to Xiang Liu; 16111017@bjtu.edu.cn

Received 24 March 2021; Accepted 21 July 2021; Published 5 August 2021

Academic Editor: Liguo Zhang

Nowadays, a lot of data mining applications, such as web traffic analysis and content popularity prediction, leverage users' web browsing trajectories to improve their performance. However, the disclosure of web browsing trajectory is the most prominent issue. A novel privacy model, named Differential Privacy, is used to rigorously protect user's privacy. Some works have applied this privacy model to spatial-temporal streams. However, these works either protect the users' activities in different places separately or protect their activities in all places jointly. The former one cannot protect trajectories that traverse multiple places; while the latter ignores the differences among places and suffers the degradation of data utility (i.e., data accuracy). In this paper, we propose a (w, n)-differential privacy to protect any spatial-temporal sequence occurring in w successive timestamps and n-range places. To achieve better data utility, we propose two implementation algorithms, named Spatial-Temporal Budget Distribution (STBD) and Spatial-Temporal RescueDP (STR). Theoretical analysis and experimental results show that these two algorithms can achieve a balance between data utility and trajectory privacy guarantee.

1. Introduction

Service providers collect users' web browsing activities and share them with many data mining applications, such as web traffic analysis and content popularity prediction. Figure 1 shows a typical data publishing system. A trusted curator aggregates users' visiting activities (a.k.a "events") and publishes the number of visits to each page periodically. The published data is a spatial-temporal stream. A user's web browsing trajectory is a sequence of timestamped visiting activities. Web browsing trajectories can reveal users' sensitive information, such as user preference. Therefore, the published stream should be protected to prevent the leakage of trajectory privacy.

A lot of excellent works [1–5] have been conducted to protect users' privacy in statistic data. Among all these works, the state-of-the-art privacy model is differential privacy [5], which can provide rigorous privacy guarantees. A differential privacy model takes a dataset as the input and outputs sanitized statistic data that remain roughly the same even if any single record in the dataset is absent. Given the output, people cannot tell whether a record is included in the original dataset or not.

Some recent works have studied the problem of releasing the spatial-temporal stream with differential privacy. According to their protection range on the time dimension, privacy models are divided into two categories: event-level privacy and user-level privacy, which are represented in Figure 1(a). Event-level privacy [6–8] protects the events on different timestamps independently. In contrast, user-level privacy [9, 10] protects users' events on all timestamps jointly. The former cannot protect the trajectories that have multiple events. The latter cannot be applied to infinite streams. The reason is that the length of a user's event sequence is infinite, which requires an infinite amount of perturbation noise. To bridge the gap between event-level privacy and user-level privacy, Kellaris et al. [11] propose the w-event differential privacy, which protects any event sequence occurring in successive w timestamps.

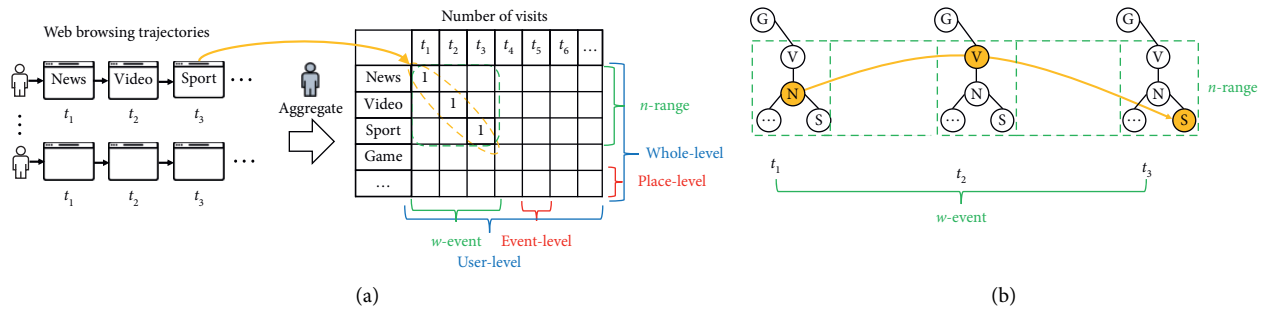


FIGURE 1: Data publishing system. (a) Event-level, user-level, place-level, and whole-level privacy and (b) a spatial-temporal window covers w timestamps and n-range places.

According to the protection range on the spatial dimension, we further categorize privacy models into two new types: place-level privacy and whole-level privacy, as shown in Figure 1(a). Place-level privacy protects the events on different places (i.e., pages) separately [10, 12, 13], whereas whole-level privacy protects the events on all places jointly [11, 14]. Since a trajectory can traverse multiple places, the place-level privacy fails to protect the trajectory privacy. However, whole-level privacy ignores the characteristics of data among different places, resulting in lower data utility (i.e., data accuracy). Specifically, the data series on different pages always have different characteristics. But whole-level privacy perturbs the data on different places with the same random noise and publishes the data on different places with the same publication cycle.

To bridge the gap between place-level privacy and whole-level privacy, it is necessary to design a new privacy model that can protect event sequences occurring in multiple places, e.g., n places.

Since the webspace is a high-dimensional space, we cannot simply arrange all pages in different rows as in Figure 1(a). Researchers usually describe the webspace as a page graph, which is shown in Figure 1(b). Intuitively, the web pages that a user browses during a short period, e.g., w successive timestamps, are related or similar. In other words, web browsing trajectories are located around a center node (e.g., topic). Therefore, in this page graph, we define that an n-range spatial window covers a center node with its $(n-1)$-hop neighbors. Given this definition, we can propose a novel privacy model, (w,n)-differential privacy, to protect any spatial-temporal sequence occurring in any spatial-temporal window of w successive timestamps and n-range paces. When $n = 1$ or N, where an N-range window covers all places, it is equivalent to place-level privacy or whole-level privacy. To provide a strong trajectory privacy guarantee, we can choose a spatial-temporal window that can cover most trajectories. Furthermore, (w,n)-differential privacy allows the data on different pages to have different perturbation noises and different publication cycles. Therefore, we can design effective implementation algorithms, which can adaptively allocate the privacy budget that controls the scale of perturbation noise and dynamically publish data according to data change. In addition to protecting web browsing trajectories, this work is also suitable for other privacy protection scenarios [15], especially non-Euclidean spaces. For example, we can use it to protect user transaction data between banks, the information flow in social networks, etc. Our contributions are summarized as follows:

(1) We demonstrate the way to construct the spatial window. Generally, the pages visited by a user within a short period, i.e., w successive timestamps, are similar or related. Therefore, we use a deep learning model to obtain the page similarity, and then we construct a page graph $\mathcal{G}$, where two similar pages are connected by an edge. In $\mathcal{G}$, we define that a spatial window of size n covers a page with its $(n-1)$-hop neighbor pages.

(2) We formulate (w,n)-differential privacy and design two implementation algorithms, Spatial-Temporal Budget Distribution (STBD) and Spatial-temporal RescueDP (STR). These two algorithms detect the data change and publish data when data change significantly. They also efficiently allocate the privacy budget for each publication to ensure that they can achieve better data utility and satisfy (w,n)-differential privacy.

(3) We compare our algorithms with two baseline algorithms, BD and RescueDP. These two algorithms belong to $(w,1)$-differential privacy and (w,N)-differential privacy, respectively. Experimental results on a real-world dataset show that the proposed two algorithms can achieve a balance between data utility and trajectory privacy guarantee.

(4) This work can be further applied to other privacy protection scenarios, especially non-Euclidean spaces, such as protecting user transaction data between banks and information flow in social networks.

The remainder of this paper is organized as follows. Section 2 surveys the related work and provides the preliminaries on differential privacy. Section 3 proposes our privacy model. Section 4 presents the STBD and STR algorithm, as well as their technical details. Section 5 presents a set of empirical studies and results. Section 6 concludes this paper.

2. Related Work and Preliminaries

2.1. Related Work

(1) Differential Privacy. Dwork et al. [5] firstly propose the differential privacy to rigorously protect individual privacy. Then, many works [16–20] apply it to the static dataset, which will not be updated in the future.

For streaming data, Dwork et al. [6] give the definitions of event-level privacy and user-level privacy and propose a "counter" algorithm to achieve event-level privacy. RTP-DMM algorithm [8] uses a Fenwick tree to reorganize data items in the stream and uses a matrix mechanism to reduce the global sensitivity. Compared with the "counter" algorithm, RTP-DMM achieves better data utility. Sun et al. [7] propose the PriStream algorithm to protect the thresholded percentile statistics under event-level privacy. Chen et al. [21] propose an event-level privacy model, PeGaSus, to simultaneously support a variety of tasks, such as counts, sliding windows, and event monitoring. To achieve user-level privacy on finite streams, Acs and Castelluccia [10] use Fourier Perturbation Algorithm [22] to perturb the data streams, and Fan and Xiong [9] propose the FAST algorithm. To bridge the gap between event-level privacy and user-level privacy, Kellaris et al. [11] propose the w-event differential privacy and its implementation algorithm, BD algorithm. Ren et al. [23] propose the average w-event differential privacy to relax the requirement of privacy budget consumed in w timestamps.

The algorithm in [10] protects the finite data series on each place separately. RescueDP [12] and AdaPub [13] independently protect the infinite data series on each place under w-event differential privacy. All these algorithms belong to the place-level privacy, which cannot rigorously protect trajectory privacy. Works [11, 14] perturb the data on different places with the same noise and publish data with the same publication cycle. Thus, they belong to whole-level privacy. Since whole-level privacy considers all places as a whole, it ignores the differences among places and achieves lower data utility. To reduce the impact of data sparsity on the spatial dimension, work [24] uses a Quadtree to group similar places together. Our previous work [25] proposes a (w, n)-differential privacy algorithm for protecting users' GPS trajectories. Because the GPS points are located in the 2D Euclidean plane, the n-range spatial window is a square area of size $n \times n$. This algorithm cannot be applied to the non-Euclidean webspace. Works [26, 27] perturb the activities within a trajectory and publish the sanitized trajectory. However, this paper aims to protect aggregated statistic data from the leakage of trajectory privacy. DADP [28] and DPCrowd [29] consider a different scenario, where the system contains multiple servers aggregating partial crowd-sourced data.

(2) Page Graph. Works [30–32] build a page graph, where the nodes are pages and the directed edges are hyperlinks. Since the hyperlinks are added by the authors, this kind of page graph cannot reflect the page relationship in users' minds. Considering users' click behaviors, Yu and Iwaihara [33] construct a click graph, where the directed and weighted edges represent click counts on each hyperlink. However, click counts often exhibit a power-law distribution [34], which makes the click graph sparse and unconnected. Inspired by recent researches on representational learning for natural language processing (NLP), Bing et al. [35] propose the session2vec algorithm extended from word2vec [36] to obtain the feature representations of pages. Their experimental results show that the cosine similarity of features can represent the relationship between any two pages.

2.2. Differential Privacy.

Differential privacy is a rigorous privacy model proposed by Dwork et al. [5]. Intuitively, given a randomized mechanism K, differential privacy requires that the output of K is insensitive to the change of a single record in the input of K. The formal definition of differential privacy is given as follows.

Definition 1. ε-differential privacy [5].

A randomized mechanism K satisfies ε-differential privacy, if for any two datasets $\mathscr{D}$ and $\mathscr{D}\prime$ differing on at most one record, and for any possible output $O \in \mathrm{Range}(K)$,

$$\Pr[K(\mathscr{D}) = O] \leq e^{\varepsilon} \times \Pr[K(\mathscr{D}\prime) = O], \tag{1}$$

where the probability is taken over the randomness of K.

The parameter ε controls the degree of privacy protection. A lower value of ε offers a stronger privacy guarantee. Two datasets $\mathscr{D}$ and $\mathscr{D}\prime$ that differ on at most one record are called neighboring datasets.

Laplace mechanism is the most commonly used mechanism that satisfies ε-differential privacy. Its main idea is to add Laplace random noise to the statistic data.

Definition 2. Sensitivity [5].

For any function $q: \mathscr{D} \longrightarrow \mathbb{R}^d$, the sensitivity of q is $\Delta(q) = \max_{\mathscr{D},\mathscr{D}\prime} \|q(\mathscr{D}) - q(\mathscr{D}\prime)\|_1$ for any two neighboring datasets $\mathscr{D}$ and $\mathscr{D}\prime$.

The sensitivity of function q is the maximum L1 distance between function outputs of any two neighboring datasets $\mathscr{D}$ and $\mathscr{D}\prime$.

Theorem 1. *Laplace mechanism [5].*

For any function $q: \mathscr{D} \longrightarrow \mathbb{R}^d$, the mechanism K

$$K(\mathscr{D}) = q(\mathscr{D}) + < \mathscr{L}_1\left(0, \frac{\triangle(q)}{\varepsilon}\right), \ldots, \mathscr{L}_d\left(0, \frac{\triangle(q)}{\varepsilon}\right) >, \tag{2}$$

satisfies ε-differential privacy, if $\mathscr{L}_i(0, \triangle(q)/\varepsilon)$ are i.i.d zero-mean Laplace noises with scale $(\triangle(q)/\varepsilon)$.

A smaller value of ε offers a larger scale of Laplace noise and a stronger privacy guarantee. Differential privacy maintains two composition properties, which are given as follows.

Theorem 2. *Sequential Composition [37].*

Let $\{K_1, K_2, \ldots, K_T\}$ be a set of mechanisms, and each K_t provides ε_t-differential privacy, where $t \in [1, T]$. Let K be another mechanism that executes $K_1(\mathscr{D})$, $K_2(\mathscr{D})$, …,

$K_T(\mathscr{D})$ independently and returns the outputs of these mechanisms. Then, K satisfies $\sum_{t=1}^{T}\varepsilon_t$-differential privacy.

Theorem 2 allows us to distribute ε among T independent mechanisms. Therefore, ε is called the privacy budget.

Theorem 3. *Parallel Composition [37].*

Let $\{D^1, D^2, \ldots, D^M\}$ be a set of disjoint subsets of dataset $\mathscr{D}$. Let $\{K^1, K^2, \ldots, K^M\}$ be a set of mechanisms, and each K^m provides ε^m-differential privacy, where $m \in [1, M]$. Let K be another mechanism that executes $K^1(D^1)$, $K^2(D^2)$, $\ldots$, $K^M(D^M)$ independently and returns the outputs of these mechanisms. Then, K satisfies $\max_{1\le m\le M}\varepsilon^m$-differential privacy.

If a dataset is partitioned into disjoint subsets, and each part is protected under differential privacy, then the ultimate privacy guarantee depends on the worst of the guarantees.

2.3. w-Event Differential Privacy. w-event differential privacy [11] (short for w-event ε-differential privacy) is used to protect any event sequence occurring in w timestamps. An infinite stream is denoted by $S = [\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_t, \ldots]$, where $\mathscr{D}_t$ denotes the dataset on timestamp t. A prefix of stream S of length T is denoted by $S_T = [\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_T]$. The definition of two w-neighboring prefixes is given as follows.

Definition 3. w-neighboring [11].

Let w be a positive integer. Two stream prefixes S_T and S_T' are w-neighboring if

(1) for each $\mathscr{D}_t, \mathscr{D}_t'$ such that $t \in [1, T]$ and $\mathscr{D}_t \neq \mathscr{D}_t'$, it holds that $\mathscr{D}_t, \mathscr{D}_t'$ are neighboring, and

(2) for each $\mathscr{D}_t, \mathscr{D}_t', \mathscr{D}_\tau, \mathscr{D}_\tau'$ with $1 \le t < \tau \le T$, $\mathscr{D}_t \neq \mathscr{D}_t'$ and $\mathscr{D}_\tau \neq \mathscr{D}_\tau'$, it holds that $\tau - t + 1 \le w$.

Concretely, if S_T, S_T' are w-neighboring prefixes, then (1) their pairwise elements $\mathscr{D}_t$ and $\mathscr{D}_t'$, where $t \in [1, T]$, are the same or neighboring, and (2) all neighboring element pairs can fit in a window of w timestamps.

Definition 4. w-event ε-differential privacy [11].

A mechanism K satisfies w-event ε-differential privacy, if, for any two w-neighboring stream prefixes S_T, S_T', any possible output $O \in \mathrm{Range}(K)$, and any $T > 0$,

$$\Pr[K(S_T) = O] \le e^{\varepsilon} \times \Pr[K(S_T') = O]. \tag{3}$$

According to Theorem 2, to satisfy w-event differential privacy, a privacy mechanism can distribute the privacy budget ε to several mechanisms that protect the data on different timestamps independently.

Theorem 4. *Implementation of w-event differential privacy [11].*

Let K be a mechanism that takes a stream prefix $S_T = [\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_T]$ as the input and outputs $O_T = [\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T] \in Range(K)$. Suppose that we can decompose K into T mechanisms $K_1, K_2, \ldots, K_T$, such that $\mathbf{o}_t = K_t(\mathscr{D}_t)$, where $t \in [1, T]$, and each K_t satisfies ε_t-differential privacy independently. Then, K satisfies w-event ε-differential privacy if

$$\sum_{t=\tau-w+1}^{\tau}\varepsilon_t \le \varepsilon, \quad \forall \tau \in [1, T]. \tag{4}$$

2.4. Dataset. In this paper, we use a real-world dataset, the WorldCup98 dataset [38]. This dataset contains 1.3 billion logs of the requests to the FIFA 1998 website from 2.76 million clients from April 30, 1998, to July 26, 1998. Each log contains client ID, URL, and timestamp. We choose 10000 most popular URLs from June 15, 1998, to June 30, 1998, and publish the number of visits every 30 minutes.

3. Privacy Model

We study the problem of how to publish the number of visits to each web page while protecting user browsing trajectories. A web browsing trajectory is defined as a sequence of web pages visited by a user with timestamps. Generally, the pages visited by a user during a short period are most likely to be similar or related. In other words, users' trajectories can fit into a spatial-temporal window that covers a group of similar pages and several successive timestamps. Therefore, to provide a strong trajectory privacy guarantee, we propose a privacy model that can protect any possible spatial-temporal sequence within a spatial-temporal window. In this section, we first construct a page graph and introduce the idea of the spatial-temporal window. Then, we formulate our privacy model.

3.1. Page Graph. Inspired by [35, 36], we use the word2vec algorithm to obtain the similarity between pages. The source code is available at [39]. We use the logs on June 15, 1998, as the training data. The embedding size, context window, and learning rate are set to 128, 3, and 0.001, respectively. After training 100000 steps with Adam optimizer [40], we obtain the feature vectors of pages. If the cosine similarity of two pages is higher than ρ, we connect them with an unweighted and undirected edge. We also connect each page to the other two pages, which are the top 2 similar pages among all pages to ensure that we can obtain a connected graph.

If ρ is too large, the generated graph will become very sparse. On the contrary, if ρ becomes too small, pages will directly connect to each other and the graph will become very dense. We test ρ from 0.1 to 0.8 and find that when $\rho = 0.4$, pages will have a proper distance to other pages.

We use $\mathscr{G}$ to denote this page graph, which contains 10000 nodes and 28594 edges. The maximum and minimum node degree are 96 and 2, respectively. Figure 2 shows the degree distribution in the log-log plot. We can observe that the distribution approximately follows the power-law [41], and the exponent of the power-law is 2.46.

The distance between two connected pages is 1, and the distance between two unconnected pages is the length of the shortest path in $\mathscr{G}$. As shown in Figure 3, the distance between page 1 and page 2 is 1. Since the shorted path from

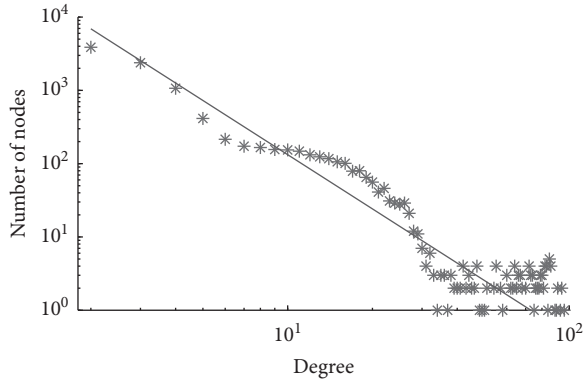
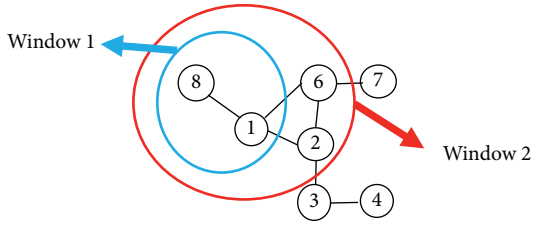


Figure 2: Degree distribution.



Trajectory A: 8-2-6
Trajectory B: 8-1-2-3-4

Figure 3: Example of the page graph.

page 1 to page 4 is $1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 4$, the distance between page 1 and page 4 is 3. We also call page 1 as a 1-hop neighborhood of page 2 and a 3-hop neighborhood of page 4. Note that a user can visit the pages that are not directly connected to each other, and then the trajectory will not be a continuous path in $\mathcal{G}$. Given the distance between pages, we define the spatial range of a trajectory as follows.

Definition 5. The spatial range of a web browsing trajectory.

Given a graph $\mathcal{G}$ and a web browsing trajectory, the center of this trajectory is a page, whose greatest distance to all pages in the trajectory is as small as possible. This distance is the spatial range of the trajectory.

Trajectory center can be regarded as the "topic" of this trajectory, and the spatial range represents the "topic scope." The trajectory center is not necessarily included in the trajectory. For example, in Figure 3, the greatest distance from page 1 to pages 8, 2, and 6 is 1, while the greatest distance from other pages to pages 8, 2, and 6 is larger than 1. Therefore, the center of trajectory A is page 1, and its spatial range is 1. Similarly, the center of trajectory B is page 2, and its spatial range is 2.

Given a page, we define that a spatial window of size $\mathbf{n}$ ($\mathbf{n}$-range spatial window) covers the page with its $(n-1)$-hop neighborhoods. As shown in Figure 3, the spatial window 1 takes page 8 as the center and covers page 8 and page 1. Similarly, the spatial window 2 takes page 1 as the center and covers page 1, page 2, page 6, and page 8. If the spatial range of a trajectory is n, this trajectory can be covered by a spatial window of size $n+1$. In $\mathcal{G}$, a spatial window of size 9 can cover all pages. A temporal window of size $\mathbf{w}$ covers w successive timestamps. We use (w, n) to denote the size of a spatial-temporal window, where the first element is the temporal size, and the second element is the spatial size.

We randomly select 50,000 users and analyze the spatial range of their trajectories during w timestamps. The interval between two consecutive timestamps is 30 minutes. The results are shown in Table 1. When $w = 120$, the spatial range of 88.7% of the trajectories is less than 5. Thus, a spatial-temporal window of 120 successive timestamps and 6-range places can cover most trajectories.

3.2. (w, n)-Differential Privacy. Without loss of generality, we assume that there are M pages hosted on a server. A user can visit at most $l \ll M$ pages per timestamp. A trusted curator collects users' visiting activities and creates a dataset $\mathcal{D}_t$ at each timestamp t. This infinite stream is denoted by $S = [\mathcal{D}_1, \mathcal{D}_2, \ldots]$. We define a stream prefix of S as $S_T = [\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T]$. Dataset $\mathcal{D}_t$ can be partitioned into M disjoint subsets $D_t^1, D_t^2, \ldots, D_t^M$, where D_t^m contains users' visiting activities to page m at timestamp t. A query function q asks for the number of visits on each page. Then, $q(\mathcal{D}_t) = [q(D_t^1), q(D_t^2), \ldots, q(D_t^M)] = \mathbf{r}_t = [r_t^1, r_t^2, \ldots, r_t^M]$, where r_t^m denotes the number of visits on page m at timestamp t. (w, n)-differential privacy protects any spatial-temporal sequence occurring in any spatial-temporal window of w timestamps and n-range places. The formal definition is given as follows.

Definition 6. (w, n)-neighboring.

Let w and n be two positive integers. Two spatial-temporal stream prefixes S_T and S_T' are (w, n)-neighboring if

(1) each pair of D_t^m, $D_t'^m$ is neighboring or the same and

(2) all neighboring pairs D_t^m, $D_t'^m$ can fit into a spatial-temporal window of size (w, n).

The above definition means that two (w, n)-neighboring stream prefixes differ on a single spatial-temporal sequence that can fit in a spatial-temporal window of size (w, n). (w, n)-differential privacy ensures that the output results of any two (w, n)-neighboring stream prefixes are indistinguishable.

Definition 7. w-event n-range ε-differential privacy.

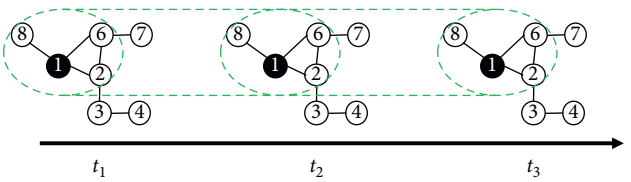
A mechanism K satisfies w-event n-range ε-differential privacy (short for (w, n)-differential privacy), if for any two (w, n)-neighboring prefixes S_T and S_T', any possible output $O \in \text{Range}(K)$, and any $T > 0$,

$$\Pr[K(S_T) = O] \leq e^{\varepsilon} \times \Pr[K(S_T') = O]. \tag{5}$$

This model can be widely used in various scenarios, where trajectories need to be protected. To implement this privacy model, we can decompose mechanism K into submechanisms K_t^m, and each K_t^m protects dataset D_t^m under ε_t^m-differential privacy independently. Then, we ensure that the total privacy budget inside any spatial-temporal window of size (w, n) is less than ε. Figure 4 gives an example. A

Table 1: Spatial range of trajectories during w timestamps.

% n	w				
	40	80	120	160	200
0	3.37	3.07	2.85	2.67	2.56
1	14.05	12.87	11.96	11.27	10.74
2	26.49	24.48	22.78	21.46	20.47
3	48.11	44.46	41.61	39.41	37.81
4	84.13	78.57	74.57	71.14	68.75
5	96.16	92.03	88.7	85.53	83.4
6	99.21	97.26	95.19	93.57	92.41
7	100	100	99.99	99.98	99.93
8	100	100	100	100	100



(a)

Page \ Time	t_1	t_2	t_3	…
1	r_1^1	r_2^1	r_3^1	
8	r_1^8	r_2^8	r_3^8	
2	r_1^2	r_2^2	r_3^2	
6	r_1^6	r_2^6	r_3^6	
…				
M				

(b)

Page \ Time	t_1	t_2	t_3	…
1	ε_1^1	ε_2^1	ε_3^1	
8	ε_1^8	ε_2^8	ε_3^8	
2	ε_1^2	ε_2^2	ε_3^2	
6	ε_1^6	ε_2^6	ε_3^6	
…				
M				

(c)

Figure 4: Example of the spatial-temporal window and privacy budget. (a) The spatial-temporal window, (b) number of visits, and (c) privacy budget.

spatial-temporal window of size $(3, 2)$ covers the data on page 1, page 2, page 6, and page 8 from t_1 to t_3. Since $r_t^1, r_t^2, r_t^6, r_t^8$ belong to disjoint datasets, the ultimate privacy guarantee is $\max(\varepsilon_t^1, \varepsilon_t^2, \varepsilon_t^6, \varepsilon_t^8)$-differential privacy based on Theorem 3. According to Theorem 2, the privacy guarantee inside this window is $\sum_{t=1}^{3}(\max(\varepsilon_t^1, \varepsilon_t^2, \varepsilon_t^6, \varepsilon_t^8))$-differential privacy. If $\sum_{t=\tau-2}^{\tau}(\max_{m\in win}\varepsilon_t^m) <= \varepsilon$ for any $\tau \in [1, T]$, and any 2-range spatial window *win*, mechanism K satisfies (3,2)-differential privacy.

Theorem 5. *Implementation of (w, n)-differential privacy.*

Let K be a mechanism that takes a spatial-temporal stream prefix $S_T = [\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_T]$ as the input and outputs $O_T = [\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T] \in Range(K)$, where $\mathscr{D}_t = [D_t^1, D_t^2, \ldots, D_t^M]$ and $\mathbf{o}_t = [o_t^1, o_t^2, \ldots, o_t^M]$. Suppose that we can decompose K into $T \times M$ mechanisms K_t^m, where $t \in [1, T], m \in [1, M]$, such that $o_t^m = K_t^m(D_t^m)$, and each K_t^m satisfies ε_t^m-differential privacy independently. Then, K satisfies (w, n)-differential privacy if

$$\sum_{t=\tau-w+1}^{\tau}\left(\max_{m\in win}\varepsilon_t^m\right)\le\varepsilon,\forall\tau\in[1,T],\quad \forall win\in\{\text{n-range spatial windows}\}. \tag{6}$$

4. Algorithms

4.1. Spatial-Temporal Budget Distribution. The structure of the Spatial-Temporal Budget Distribution (STBD) algorithm is shown in Figure 5. STBD consists of three components: evaluation, perturbation, and budget allocation. The evaluation component evaluates the data change and decides whether to publish new data at each timestamp. The perturbation component perturbs the data with Laplace noise and publishes the sanitized data. The budget allocation component distributes the privacy budget to each component. Our STBD algorithm is extended from the Budget Distribution (BD) algorithm [11]. STBD uses a new evaluation component to capture the data change on small page groups and uses a new budget allocation component to satisfy (w,n)-differential privacy. Algorithm 1 shows the pseudocode of the STBD algorithm.

4.1.1. Evaluation and Perturbation. Publishing data consumes the privacy budget, while the total privacy budget inside the spatial-temporal window is constant. If we publish data on every timestamp, the privacy budget for each publication becomes small, which results in significant perturbation noise. Conversely, if we skip some publications and approximate current data to previously published data, the perturbation noise decreases, while the approximation error increases. The approximation error is calculated as follows:

$$a\_\mathrm{err}_t^m=\left|r_t^m-o_{t-1}^m\right|+\mathscr{L}\left(0,\frac{l}{\alpha_t^m}\right), \tag{7}$$

r_t^m denotes the number of visits to page m at timestamp t. o_{t-1}^m denotes the previously published data on page m at timestamp $t-1$. Because the approximation error contains the sensitive data r_t^m, we should add Laplace noise to it. Since one user can visit at most l pages per timestamp, the sensitivity of the approximation error is l. α_t^m is the privacy budget for evaluation. $\mathscr{L}(0,l/\alpha_t^m)$ is called the evaluation noise.

The perturbation component adds perturbation noise to data r_t^m. The sanitized output data is given as follows:

$$o_t^m=r_t^m+\mathscr{L}\left(0,\frac{l}{\beta_t^m}\right), \tag{8}$$

β_t^m is the privacy budget for perturbation. Because the perturbation noise is a random variable, we define the perturbation error as the expectation of the absolute error between r_t^m and o_t^m. The perturbation error is calculated as follows:

$$p\_\mathrm{err}_t^m=E\left(\left|r_t^m-o_t^m\right|\right)=E\left(\left|\mathscr{L}\left(0,\frac{l}{\beta_t^m}\right)\right|\right)=\frac{l}{\beta_t^m}. \tag{9}$$

The evaluation component compares these two errors and decides whether to publish new data. If the data change dramatically, the approximation error becomes larger than the perturbation error. Then, the perturbation component consumes the perturbation budget and publishes new data. Otherwise, it skips this publication and saves the perturbation budget.

Unfortunately, the evaluation noise can easily cause the evaluation component to make the bad decisions. To reduce the evaluation noise, we cluster all pages into X groups and calculate the mean approximate error in each group instead of the approximate error on each page. The mean approximate error is calculated as follows:

$$\overline{a\_\mathrm{err}_t^x}=\frac{1}{\mathrm{size}(x)}\sum_{m\in x}\left|r_t^m-o_{t-1}^m\right|+\mathscr{L}\left(0,\frac{l/\mathrm{size}(x)}{\alpha_t^x}\right), \tag{10}$$

where $\mathrm{size}(x)$ stands for the number of pages in group $x\in[1,X]$. The sensitivity of the mean approximate error is $l/\mathrm{size}(x)$, and α_t^x is the evaluation budget for each group.

We can see that the evaluation noise in equation (10) has a smaller variance compared to equation (7). However, the mean approximate error also has a shortcoming that the data change on a single page could be ignored. BD algorithm calculates the mean approximate error on all pages, which ignores many small changes among pages and results in a lower data utility. STBD clusters pages into several groups to strike a balance between reducing evaluation noise and capturing data change. The pseudocode of the evaluation and the perturbation are shown in Lines 2–18, Algorithm 1. The evaluation component first calculates the approximation error (Lines 2–7). Then, it calculates the perturbation error (Lines 8–10). If the perturbation error is less than the approximation error, the perturbation component adds perturbation noise to r_τ^m and publishes it (Lines 11–13). Otherwise, the perturbation component repeats the previously published data $o_{\tau-1}^m$ and saves the perturbation budget (Lines 14–17).

4.1.2. Budget Allocation in STBD. The budget allocation component distributes half of the privacy budget ε to the evaluation component, and the other half to the perturbation component. The evaluation budget is evenly distributed to w timestamps. Therefore, the evaluation budget $\alpha_t^x=(\varepsilon/2)/w$ for any timestamp t and any page group x. The perturbation budget for each publication is allocated by an exponential decay method, which allocates half of the remaining budget to each publication. Algorithm 2 presents the pseudocode of budget allocation.

We use Figure 6 as an example to illustrate Algorithm 2. Windows 1 and 2 are two 2-range spatial windows that cover page 8. Windows 1 and 2 take pages 8 and 1 as the center, respectively. The temporal length of windows 1 and 2 is w. Considering window 1, the perturbation budget consumed at timestamp t is calculated as follows:

$$\phi_t^{win1}=\max_{m\in win1}\beta_t^m. \tag{11}$$

Input: $\mathbf{r}_\tau, \mathbf{o}_{\tau-1}$
Output: $\mathbf{o}_\tau, (\beta_\tau^1, \ldots, \beta_\tau^M)$
(1) $\alpha_\tau^x, \beta_\tau^m \leftarrow$ Budget Allocation
//Evaluation
(2) for group x do
(3) $\overline{a\_\text{err}_\tau^x} = (1/\text{size}(x))\sum_{m\in x}|r_\tau^m - o_{\tau-1}^m| + \mathscr{L}(0, (l/\text{size}(x))/\alpha_\tau^x)$
(4) for page $m \in x$ do
(5) $a\_\text{err}_\tau^m = \overline{a\_\text{err}_\tau^x}$
(6) end for
(7) end for
(8) for page m do
(9) $p\_\text{err}_\tau^m = l/\beta_\tau^m$
(10) end for
//Perturbation
(11) for page m do
(12) if $a\_\text{err}_\tau^m > p\_\text{err}_\tau^m$ then
(13) $o_\tau^m = r_\tau^m + \mathscr{L}(0, l/\beta_\tau^m)$
(14) else
(15) $o_\tau^m = o_{\tau-1}^m$
(16) $\beta_\tau^m = 0$
(17) end if
(18) end for
(19) return $\mathbf{o}_\tau, (\beta_\tau^1, \ldots, \beta_\tau^M)$

Algorithm 1: STBD algorithm.

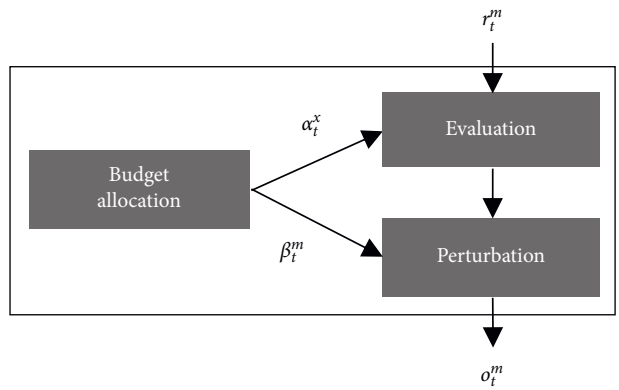
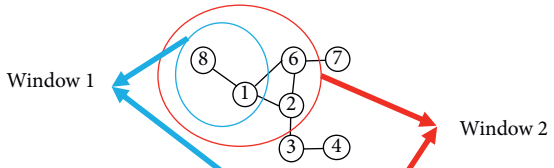


Figure 5: The structure of the STBD algorithm.

Input: $(\beta_1^1, \ldots, \beta_{\tau-1}^M), \varepsilon, w, n$
Output: $\alpha_\tau^x, \beta_\tau^m$
(1) for group x do
(2) $\alpha_\tau^x = (\varepsilon/2)/w$
(3) end for
(4) for n-range window win do
(5) $\phi_t^{win} = \max_{m\in win} \beta_t^m$
(6) $\psi_\tau^{win} = \varepsilon/2 - \sum_{t=\tau-w+1}^{\tau-1} \phi_t^{win}$
(7) end for
(8) for node m do
(9) $\eta_\tau^m = \min_{win\in m} \psi_\tau^{win}$
(10) $\beta_\tau^m = \eta_\tau^m/2$
(11) end for
(12) return $\alpha_\tau^x, \beta_\tau^m$

Algorithm 2: Budget Allocation in STBD.



Page \ Time	$\tau - w + 1$	…	$\tau - 1$
1	$\beta^1_{\tau-w+1}$	…	$\beta^1_{\tau-1}$
8	$\beta^8_{\tau-w+1}$	…	$\beta^8_{\tau-1}$
2	$\beta^2_{\tau-w+1}$	…	$\beta^2_{\tau-1}$
6	$\beta^6_{\tau-w+1}$	…	$\beta^6_{\tau-1}$

Window \ Time	$\tau - w + 1$	…	$\tau - 1$
1	$\phi^{win1}_{\tau-w+1}$	…	$\phi^{win1}_{\tau-1}$
2	$\phi^{win2}_{\tau-w+1}$	…	$\phi^{win2}_{\tau-1}$

Figure 6: Illustration of exponential decay method.

The remaining budget of window 1 at timestamp τ is

$$\psi^{win1}_{\tau} = \frac{\varepsilon}{2} - \sum_{t=\tau-w+1}^{\tau-1} \phi^{win1}_{t}. \tag{12}$$

Similarly, the remaining budget of window 2 at timestamp τ is

$$\psi^{win2}_{\tau} = \frac{\varepsilon}{2} - \sum_{t=\tau-w+1}^{\tau-1} \phi^{win2}_{t}. \tag{13}$$

To ensure that the perturbation budget inside windows 1 and 2 is no more than $\varepsilon/2$, the remaining budget for page 8 at timestamp τ is

$$\eta^{8}_{\tau} = \min_{win \ni 8} \psi^{win}_{\tau}. \tag{14}$$

We allocate half of the remaining budget for the perturbation budget, while reserving the other half for future publications. The perturbation budget is described as follows:

$$\beta^{8}_{\tau} = \frac{1}{2}\eta^{8}_{\tau}. \tag{15}$$

4.1.3. Privacy Analysis

Theorem 6. *STBD algorithm satisfies the (w, n)-differential privacy.*

Proof. Given a page m belonging to group x, the evaluation component calculates the mean approximate error on group x and the perturbation error on page m. The perturbation error does not contain sensitive information. The sensitivity of the mean approximate error on group x is $l/\text{size}(x)$. Based on Theorem 1, evaluation component adds $\mathscr{L}(0, (l/\text{size}(x))/\alpha^x_t)$ to the mean approximate error, so it satisfies α^x_t-differential privacy.

The perturbation component adds $\mathscr{L}(0, l/\beta^m_t)$ to the data r^m_t and the sensitivity of r^m_t is l. According to Theorem 1, the perturbation component satisfies β^m_t-differential privacy.

The evaluation and perturbation protect the data r^m_t independently. Therefore, r^m_t is protected under $(\alpha^x_t + \beta^m_t)$-differential privacy based on Theorem 2. Budget allocation guarantees that $\sum_{t=\tau-w+1}^{\tau} \max_{m \in win}(\alpha^x_t + \beta^m_t) \le \varepsilon$ for any n-range window win and any τ, so STBD algorithm satisfies (w, n)-differential privacy. □ □

4.2. Spatial-Temporal RescueDP. Spatial-Temporal RescueDP (STR) extends the RescueDP [12] to implement (w, n)-differential privacy. The structure of the STR algorithm is shown in Figure 7. It contains four components: sampling, perturbation, filtering, and budget allocation. The sampling component adaptively selects the timestamps to publish new data according to the data change and the remaining budget. The perturbation component adds Laplace noise to the statistic data at sampling timestamps. The filtering component uses the Kalman Filter [9, 12] to improve data utility. The budget allocation component allocates the privacy budget for each publication. The difference between STR and RescueDP is that STR uses a new budget allocation component to achieve (w, n)-differential privacy. Algorithm 3 outlines the steps in STR.

4.2.1. Sampling. The sampling component uses the PID control algorithm [12] to predict the data change. The PID error is defined as follows:

$$\begin{aligned} E^m_{t_n} &= \left| o^m_{t_n} - o^m_{t_{n-1}} \right|, \\ \delta^m_{t_n} &= K_p E^m_{t_n} + K_g \frac{\sum_{u=n-\pi-1}^{n} E^m_{t_u}}{\pi} + K_d \frac{E^m_{t_n}}{t_n - t_{n-1}}. \end{aligned} \tag{16}$$

The subscript t_n indicates the n-th sampling timestamp. $E^m_{t_n}$ is the feedback error [12], which calculates the absolute error between the published data at the current sampling timestamp and the last sampling timestamp. PID error $\delta^m_{t_n}$ contains three parts: (1) $K_p E^m_{t_n}$ is the proportional error standing for the current error; (2) $K_g \sum_{u=n-\pi-1}^{n} E^m_{t_u}/\pi$ is the integral error standing for the sum of past π feedback errors; (3) $K_d E^m_{t_n}/(t_n - t_{n-1})$ is the derivative error standing for the future error.

When the data change rapidly, the PID error becomes larger. To reduce approximation error, the sampling interval should become smaller. Meanwhile, if the remaining budget is small, the sampling interval should become larger to reduce the perturbation noise. Therefore, the sampling interval is calculated as follows:

Input: $\mathbf{r}_\tau$
Output: $\mathbf{o}_\tau$
(1) Find the set of sampling pages
(2) Budget Allocation component allocates privacy budget to each page
(3) Perturbation component adds Laplace noise to r_τ^m
(4) Filtering component uses Kalman filter to reduce the noise
(5) Sampling component calculates the next sampling interval

Algorithm 3: STR algorithm.

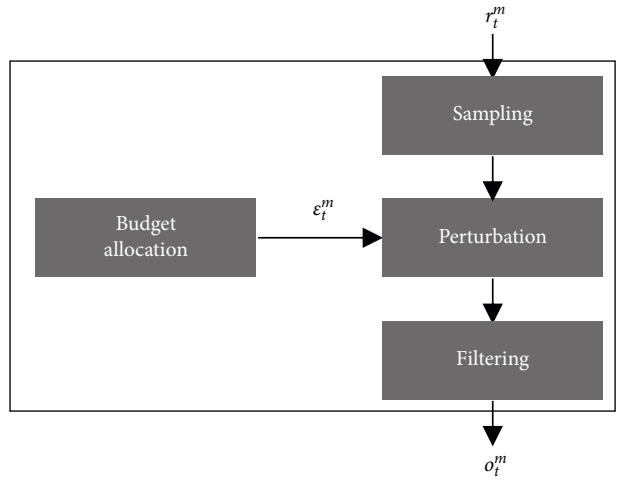


Figure 7: The structure of the STR algorithm.

$$I_{t_n}^m = \max\left(1, I_{t_{n-1}} + \theta\left(1 - \left(\delta_{t_n}^m \cdot \eta_{t_n}^m\right)^2\right)\right), \tag{17}$$

$I_{t_n}^m$ is the sampling interval of page m at timestamp t_n. $\eta_{t_n}^m$ is the remaining budget for page m at timestamp t_n. The parameter θ controls the changing rate of the sampling interval.

4.2.2. Perturbation and Filtering. The perturbation component adds the Laplace random noise into data r_t^m. The perturbation budget is denoted by ε_t^m. The sanitized output data is calculated as follows:

$$o_t^m = r_t^m + \mathscr{L}\left(0, \frac{l}{\varepsilon_t^m}\right). \tag{18}$$

The filtering component uses the Kalman Filter to reduce the perturbation noise. Since it only accesses the sanitized output data, it does not leak sensitive information. More details of Kalman Filter can be found in [9, 12].

4.2.3. Budget Allocation in STR. Budget allocation in STR allocates the perturbation budget based on the sampling interval. If the sampling interval is small, the number of sampling timestamps could be more. Therefore, we should allocate a small portion of the remaining budget to the current timestamp and leave more budget to the future. The pseudocode is shown in Algorithm 4. It first calculates the remaining budget for each page (Lines 1–6). Then, it calculates the proportion p of the remaining budget allocated to the perturbation component (Line 7). The $p_{\max}$ and $\varepsilon_{\max}$ are used to avoid consuming too much budget at a single publication (Lines 7-8).

4.2.4. Privacy Analysis

Theorem 7. *STR algorithm satisfies the (w, n)-differential privacy*

Proof. The perturbation component accesses the sensitive data r_t^m, while the other three components do not. The perturbation component adds $\mathscr{L}(0, l/\varepsilon_t^m)$ to the data r_t^m and the sensitivity of r_t^m is l. According to Theorem 1, the perturbation component satisfies ε_t^m-differential privacy. The budget allocation component guarantees that $\sum_{t=\tau-w+1}^{\tau} \max_{m \in win} \varepsilon_t^m < \varepsilon$ for any n-range window win and any τ. Therefore, STR algorithm satisfies (w, n)-differential privacy. □ □

5. Experimental Evaluation

Baseline models. In this section, we compare STBD and STR with two state-of-the-art algorithms, BD [11] and RescueDP [12]. BD is a (w, N)-differential privacy algorithm, and RescueDP is a $(w, 1)$-differential privacy

Input: $(\varepsilon_1^1, \ldots, \varepsilon_{\tau-1}^M), \varepsilon, w, n$
Output: ε_τ^m
(1) for n-range window win do
(2) $\quad \phi_t^{win} = \max_{m \in win} \varepsilon_t^m$
(3) $\quad \psi_\tau^{win} = \varepsilon - \sum_{t=\tau-w+1}^{\tau-1} \phi_t^{win}$
(4) end for
(5) for node m do
(6) $\quad \eta_\tau^m = \min_{win \in m} \psi_\tau^{win}$
(7) $\quad p_\tau^m = \min(s \cdot ln(I_\tau^m + 1), p_{\max})$
(8) $\quad \varepsilon_\tau^m = \min(p_\tau^m \cdot \eta_\tau^m, \varepsilon_{\max})$
(9) end for
(10) return ε_τ^m

Algorithm 4: Budget allocation in STR.

Table 2: STR parameters.

K_p	0.9	θ	10
K_g	0.1	s	0.2
K_d	0	$p_{\max}$	0.6
π	3	$\varepsilon_{\max}$	0.2ε

algorithm. Since the Uniform algorithm [11], which allocates the same budget to every timestamp, performs much worse than baseline models, we do not include it in the following experiments. All algorithms are written by Python on a computer with Intel Core i7-8700 CPU. We run each experiment 50 times and report the average results.

Utility metrics. We use the Mean Absolute Error (MAE) and Mean Relative Error (MRE) as the utility metrics to measure the performance of algorithms.

$$\begin{aligned} \mathrm{MAE}(R_T, O_T) &= \frac{1}{T \cdot M} \sum_{t=1}^{T} \sum_{m=1}^{M} \left| r_t^m - o_t^m \right|, \\ \mathrm{MRE}(R_T, O_T) &= \frac{1}{T \cdot M} \sum_{t=1}^{T} \sum_{m=1}^{M} \frac{\left| r_t^m - o_t^m \right|}{\max(\gamma, r_t^m)}, \end{aligned} \tag{19}$$

where $R_T = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_T]$ denotes the real number of visits, and $O_T = [\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T]$ is the sanitized output. γ is used to avoid the special case when r_t^m is 0. We set γ to 0.1% of $\sum_{m=1}^{M} r_t^m$, which is the same as [12].

In the STBD algorithm, we use the Louvain algorithm [42] to cluster pages into 39 groups. To fairly compare with RescueDP, STR uses the same parameters in [12], which are shown in Table 2. We refer readers to [12] for more details. We set l to 20.

Utility vs. ε. Figure 8 shows the MAE and MRE of four algorithms when ε changes from 0.2 to 1.0 under the condition that $w = 120$ and $n = 6$. Because larger privacy budget results in smaller Laplace random noise, the MAE and MRE of the four algorithms decrease when ε becomes larger. RescueDP and STR outperform BD and STBD. Since BD and STBD only allocate half of the privacy budget for the perturbation component, they get larger perturbation noise. Furthermore, RescueDP and STR use Kalman Filter to reduce the random noise.

Utility vs. w. Figure 9 shows the MAE and MRE of four algorithms when w changes from 40 to 200 under the condition that $\varepsilon = 1$ and $n = 6$. The MAE and MRE of BD and STBD increase when w becomes larger. This is because the evaluation budget becomes smaller when w becomes larger, which results in larger evaluation noise and more bad decisions. Although a larger w may result in more publications and less perturbation budget for each publication, STR and RescueDP increase the sampling interval when the remaining budget becomes small. Therefore, the perturbation budget cannot be too small, and the performance of STR and RescueDP is relatively stable when w changes.

Utility vs. n. Figure 10 shows the MAE and MRE of four algorithms when n changes from 4 to 9 under the condition that $\varepsilon = 1$ and $w = 120$. Because the n-range spatial window constrains the maximum perturbation budget consumed on each page, the performance of STR is worse than RescueDP. We can also observe that the performance of STBD and STR is stable as n increases. The reason is that STBD and STR can prevent performance degradation when n becomes too large. A much larger n brings a stronger budget constraint on the spatial dimension and can lead to greater perturbation noise. However, STBD and STR evaluate the data changes and skip some publications to save the budget for future publications. Thus, the performance degradation is not obvious. On the contrary, if we publish the data at every timestamp, data utility will significantly decrease as n increases.

Utility vs. Trajectory privacy guarantee. Comparing four algorithms in Figures 8–10, we can observe that BD gets the worst data utility. However, BD protects any spatial-temporal sequence within w successive timestamps, so it provides the strongest trajectory privacy guarantee. RescueDP

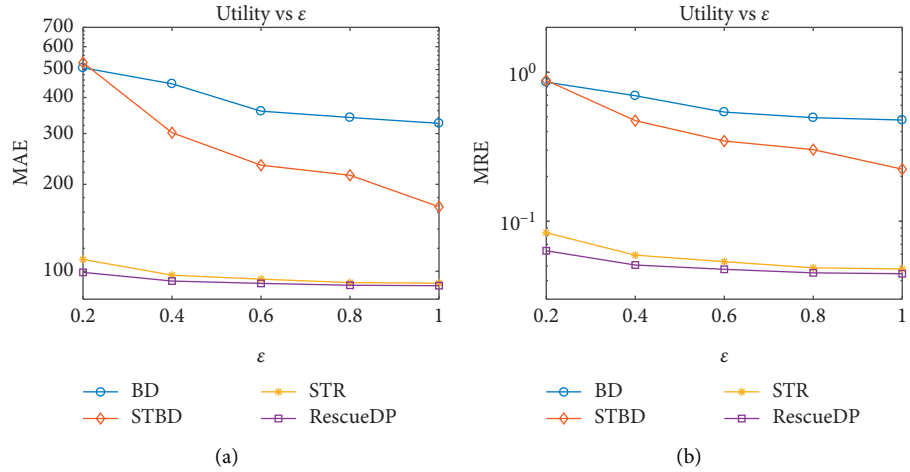


Figure 8: Utility vs. ε.

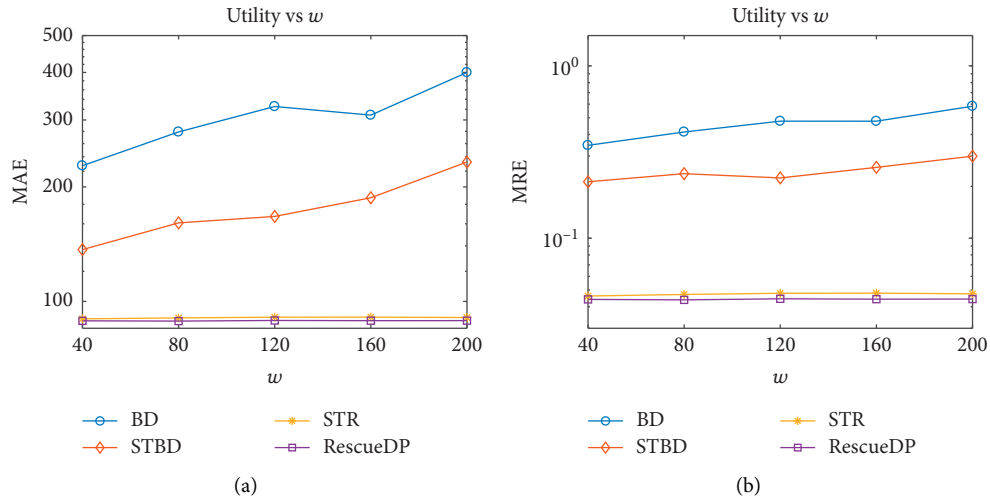


Figure 9: Utility vs. w.

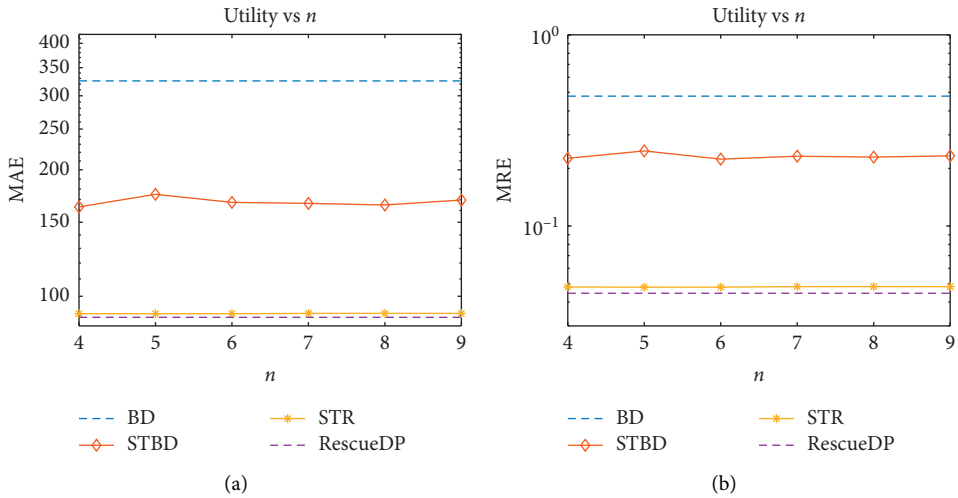


Figure 10: Utility vs. n.

gets the best data utility, but it cannot protect trajectories that traverse multiple places. STBD allows pages to have different publication decisions and different perturbation noises. Thus, STBD outperforms BD. STR constrains the perturbation budget consumed by the pages in the n-range spatial window. Therefore, the performance of STR is worse than that of RescueDP. STBD and STR protect any spatial-temporal sequence within any spatial-temporal window of size (w, n), which can cover most trajectories. Therefore, STBD and STR can achieve a balance between data utility and trajectory privacy guarantee.

6. Conclusion

In this paper, we propose (w, n)-differential privacy to protect trajectory privacy in spatial-temporal streams. This privacy model protects any spatial-temporal sequence occurring in any window of w timestamps and n-range places. We introduce the way of constructing the spatial-temporal window and finding the appropriate window size. To achieve better data utility, two implementation algorithms, STBD and STR, are proposed. Both of these two algorithms adaptively allocate the privacy budget and publish data according to the characteristics of data. Experiments on a real-world dataset show that our proposed algorithms can achieve a balance between data utility and trajectory privacy guarantee.

Data Availability

The pseudocode of our algorithm is given in the article. The web browsing history and source code of word2vec used to support the findings of this study are included within the references.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 61572071 and 61872031 and the Fundamental Research Funds for the Central Universities under Grant no. 2019YJS020.

References

[1] L. Sweeney, "k-ANONYMITY: a model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[2] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: privacy beyond k-anonymity," in *Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*, p. 24, April 2006.

[3] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: privacy beyond k-anonymity and l-diversity," in *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, IEEE, Istanbul, Turkey, April 2007.

[4] M. Prakash and G. Singaravel, "A new model for privacy preserving sensitive data mining," in *Proceedings of the 2012 Third International Conference on Computing, Communication and Networking Technologies*, pp. 1–8, ICCCNT'12), Karur, India, July 2012.

[5] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the Theory of Cryptography Conference*, pp. 265–284, Springer, New York, NW, USA, March 2006.

[6] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, pp. 715–724, ACM, Cambridge, Massachusetts, June 2010.

[7] J. Sun, R. Zhang, J. Zhang, and Y. Zhang, "Pristream: privacy-preserving distributed stream monitoring of thresholded percentile statistics," in *Proceedings of the IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, IEEE, San Francisco, CA, USA, April 2016.

[8] L. Sun, C. Ge, X. Huang, Y. Wu, and Y. Gao, "Differentially private real-time streaming data publication based on sliding window under exponential decay," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 61–78, 2019.

[9] L. Fan and L. Xiong, "Real-time aggregate monitoring with differential privacy," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pp. 2169–2173, ACM, Maui, HI, USA, October 2012.

[10] G. Acs and C. Castelluccia, "A case study: privacy preserving release of spatio-temporal density in paris," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1679–1688, ACM, New York, NW, USA, August 2014.

[11] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, "Differentially private event sequences over infinite streams," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1155–1166, 2014.

[12] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, "Real-time and spatio-temporal crowd-sourced social network data publishing with differential privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, p. 1, 2016.

[13] T. Wang, X. Yang, X. Ren, J. Zhao, and K.-Y. Lam, "Adaptive differentially private data stream publishing in spatio-temporal monitoring of iot," in *Proceedings of the 2019 IEEE 38th International Performance Computing and Communications Conference*, pp. 1–8, IPCCC), London, England, October 2019.

[14] L. Fan, L. Bonomi, L. Xiong, and V. Sunderam, "Monitoring web browsing behavior with differential privacy," in *Proceedings of the 23rd International Conference on World Wide Web*, pp. 177–188, ACM, Seoul, Korea, April 2014.

[15] X. Yang, T. Wang, X. Ren, and W. Yu, "Survey on improving data utility in differentially private sequential data publishing," *IEEE Transactions on Big Data*, p. 1, 2017.

[16] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett, "Differentially private histogram publication," *The VLDB Journal*, vol. 22, no. 6, pp. 797–822, 2013.

[17] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, Arlington, VA, USA, April 2012.

[18] W. Qardaji, W. Weining Yang, and N. Ninghui Li, "Differentially private grids for geospatial data," *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, ICDE), in *Proceedings of the 2013 IEEE 29th International Conference on Data Engineering*, pp. 757–768, April 2013.

[19] Y. Xia, T. Zhu, X. Ding, H. Jin, and D. Zou, "Heterogeneous differential privacy for vertically partitioned databases," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 8, p. e5607, 2019.

[20] D. Lv and S. Zhu, "Achieving correlated differential privacy of big data publication," *Computers & Security*, vol. 82, pp. 184–195, 2019.

[21] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau, "Pegasus: data-adaptive differentially private stream processing, CCS '17," in *Proceedings of the Association for Computing Machinery*, pp. 1375–1388, New York, NY, USA, June 2017.

[22] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pp. 735–746, ACM, Indianapolis, , IND, USA, June 2010.

[23] X. Ren, S. Wang, X. Yao, C.-M. Yu, W. Yu, and X. Yang, "Differentially private event sequences over infinite streams with relaxed privacy guarantee," in *Wireless Algorithms, Systems, and Applications Wireless Algorithms, Systems, and Applications*, E. S. Biagioni, Y. Zheng, and S. Cheng, Eds., , 2019.

[24] L. Fan, L. Xiong, and V. Sunderam, "Differentially private multi-dimensional time series release for traffic monitoring," in *Lecture Notes in Computer Science Data and Applications Security and Privacy XXVII*, L. Wang and B. Shafiq, Eds., Springer Berlin Heidelberg, Berlin, Germany, 2013.

[25] X. Liu, Y. Guo, Y. Chen, and X. Tan, "Trajectory privacy protection on spatial streaming data with differential privacy," in *Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, IEEE, Abu Dhabi, United Arab Emirates, December 2018.

[26] Y. Cao, Y. Xiao, L. Xiong, L. Bai, and M. Yoshikawa, "PriSTE," *Proceedings of the VLDB Endowment*, vol. 12, no. 12, pp. 1866–1869, 2019.

[27] H. Huang, X. Niu, C. Chen, and C. Hu, "A differential private mechanism to protect trajectory privacy in mobile crowd-sensing," in *Proceedings of the 2019 IEEE Wireless Communications and Networking Conference*, pp. 1–6, WCNC), Marrakesh, Morocco, April 2019.

[28] Z. Wang, X. Pang, Y. Chen et al., "Privacy-preserving crowd-sourced statistical data publishing with an untrusted server," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1356–1367, 2019.

[29] X. Ren, C.-M. Yu, W. Yu, X. Yang, J. Zhao, and S. Yang, "Dpcrowd: privacy-preserving and communication-efficient decentralized statistical estimation for real-time crowd-sourced data," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2775–2791, 2021.

[30] R. . Albert, H. Jeong, and L. A. Barabási, "Diameter of the world wide web," *Nature*, vol. 401, no. 6, pp. 130-131, 1999.

[31] A. Broder, R. Kumar, F. Maghoul et al., "Graph structure in the web," *Computer Networks*, vol. 33, no. 1-6, pp. 309–320, 2000.

[32] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web," Stanford InfoLab, Stanford, CA, USA, Tech. rep, 1999.

[33] L. Yu and M. Iwaihara, "Finding high quality documents through link and click graphs," *2018 7th International Congress on Advanced Applied Informatics (IIAI-AAI)*, IIAI-AAI), in *Proceedings of the 2018 7th International Congress on Advanced Applied Informatics*, pp. 49–54, July 2018.

[34] M. Naldi, "Approximation of the truncated zeta distribution and zipf 's law," 2015.

[35] L. Bing, Z. Y. Niu, P. Li, W. Lam, and H. Wang, "Learning a unified embedding space of web search from large-scale query log," *Knowledge-Based Systems*, vol. 150, Article ID S095070511830100X, 2018.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *Computer Science*.

[37] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, pp. 19–30, ACM, New York, NW, USA, June 2009.

[38] B. Lab, "Worldcup1998," 1998, http://ftp://ita.ee.lbl.gov/html/contrib/WorldCup.html.

[39] "Google, word2vec," 2019, https://github.com/tensorflow/tensorflow/tree/r0.11/tensorflow/examples/tutorials/word2vec.

[40] D. Kingma, J. Ba, and Adam, "A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations*, p. 12, Banff, Canada, April 2014.

[41] M. Newman, "Power laws, Pareto distributions and Zipf's law," *Contemporary Physics*, vol. 46, no. 5, pp. 323–351, 2005.

[42] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and experiment*, vol. 2008, no. 10, Article ID P10008, 2008.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9997641, 11 pages
https://doi.org/10.1155/2021/9997641

Research Article

A Vulnerability Detection System Based on Fusion of Assembly Code and Source Code

Xingzheng Li, Bingwen Feng, Guofeng Li, Tong Li, and Mingjin He

College of Information Science and Technology, Jinan University, Guangzhou 510632, China

Correspondence should be addressed to Bingwen Feng; bingwfeng@gmail.com

Received 15 March 2021; Accepted 17 July 2021; Published 30 July 2021

Academic Editor: Liguo Zhang

Software vulnerabilities are one of the important reasons for network intrusion. It is vital to detect and fix vulnerabilities in a timely manner. Existing vulnerability detection methods usually rely on single code models, which may miss some vulnerabilities. This paper implements a vulnerability detection system by combining source code and assembly code models. First, code slices are extracted from the source code and assembly code. Second, these slices are aligned by the proposed code alignment algorithm. Third, aligned code slices are converted into vector and input into a hyper fusion-based deep learning model. Experiments are carried out to verify the system. The results show that the system presents a stable and convergent detection performance.

1. Introduction

Software vulnerability detection is crucial to cybersecurity defenses. In recent years, the number of software vulnerabilities reported has grown rapidly with the development of the software industry. According to Common Vulnerabilities and Exposures (CVE) [1], 6447 security vulnerabilities were published in 2016, and this number has increased to 9000 in the first half of 2020. These vulnerabilities have led to many cybersecurity incidents. An effective tool to handle these software vulnerabilities is vulnerability detection. For this reason, many scholars and businesses have devoted much of their energies to the research of vulnerability detection.

There are a wide variety of vulnerability detection methods for discovering vulnerabilities in software. They can be broadly divided into dynamic analysis and static analysis methods. The former identify vulnerable behaviors in the process of analyzing and executing software [2, 3]. Despite a low false alarm rate, this type of methods may miss some vulnerabilities due to the difficulty of analyzing the entire program's behavior [4]. The latter detect software vulnerabilities by analyzing a code without executing it. They have high convergence and, thus, may be more popular than the former. Some static analysis-based detectors explore code similarity to detect the vulnerabilities caused by code cloning [5–7]. However, these approaches are hard to generalize to other types of vulnerabilities. In contrast, pattern-based, especially machine learning-based, detectors can use a large number of software codes to learn the patterns of universal vulnerabilities [8, 9]. Recently, deep learning models have been introduced to extract vulnerability features from program fragments [10–12] and achieved considerable accuracy.

The method based on code metrics [13] selects appropriate metrics to predict vulnerabilities. This method can complete the detection quickly, but with a low accuracy. The graph-based deep learning vulnerability detection method in [14] extracts the graph of the code to predict vulnerabilities. This method has a high detection effectiveness; however, the detection speed is too slow for large-scale code detection. In order to balance the detection speed and accuracy, we use a similar token sequence-based deep learning model as those in [10–12, 15]. These methods extract the token sequences related to the vulnerability information. Some of them extract token sequences from source codes [10, 11], while some use assembly codes. Experimentally we find that, due to the limitations of employing a single model, these schemes do not perform very stably when facing some vulnerabilities, thus reducing their performances. In view of this, we

combine the source code model and assembly code model to enhance the detection ability.

Multimodal machine learning has the ability of processing and understanding multisource modal information. Multimodal fusion is one of the earliest concerns for multimodal machine learning. It processes data from different modalities and obtains more characteristics by extracting multimodal data. Multimodal fusion usually achieves better results than only using a single mode. In audio-visual speech recognition (AVSR), the scheme in [16] fuses visual and audio cues to improve the performance relative to audio-only recognition. The scheme in [17] uses video features fusion with spectral and prosodic speech information for multimodal laughter detection. Compared to using only video features, performance can be improved by up to 3.7%. In multimodal emotion recognition, the scheme in [18] considers video only, audio only, and audio visual for the purpose of emotion recognition. The scheme in [19] uses the early fusion of information from facial expressions, body movements, gestures, and speech to recognize emotions. These methods successfully improved the recognition rate of emotions on the original basis. As a result, we believe that the multimodal fusion of code information could also be usable for vulnerability detection.

In this paper, a vulnerability detection system is proposed by fusing the assembly code and source code models (code and data are available: https://github.com/onstar99/VulnerabilitySystem). Given the source code of a software program, it is compiled to obtain the corresponded assembly code. Then, code slices are extracted from both the source code and the assembly code. After that, a code alignment algorithm is suggested to connect each source code statement with its corresponded assembly code statements. Then, the aligned codes are combined to form a fused slice. These new slices, together with the original source code slices and assembly code slices, are fed into the hyper fusion-based deep learning model for vulnerability detection. The main contributions of this paper include the following:

(1) We improve the performance of the vulnerability detection by fusing the models of assembly codes and source codes

(2) We suggest a simple but effective alignment method between source codes and assembly codes to quickly align the data slices

(3) We collect a vulnerability dataset composed of source and assembly codes, which can be used to train and verify the proposed multimodal-based vulnerability detection method

2. Related Work

2.1. Code Representation for Vulnerability Detection. In order to better express vulnerability characteristics in a deep learning-based vulnerability detector, code representation is usually introduced as an intermediate representation. Code representation commonly used includes code metrics, token sequences, abstract syntax trees, and graphs. Token sequences are directly obtained by statistics at the code level, so they have a strong correlation with vulnerabilities. The scheme in [23] scans the character stream of a code and marks its important information according to the lexical rules of a programming language. Then, the character stream of the code is converted into a token sequence representation. Finally, through vectorization processing such as one-hot and word2vec [24], vectors can be obtained and used as the input of a machine learning model.

2.2. Vulnerability Detection by Source Code. Source codes are computer program files that have not yet been compiled. Many vulnerability detectors are implemented by parsing the source code. Li et al. [10] extracted Syntax-based Vulnerability Candidates (SyVCs) from the source code and converted it to Semantics-based Vulnerability Candidates (SeVCs) through a suggested algorithm. Then, word2vec [24] was used to convert SeVC into vectors that facilitate deep learning. Li et al. [11] used LLVM IR technique and suggested a fine-grained deep learning-based vulnerability detector to locate the vulnerability accurately. Cao et al. [20] extracted abstract syntax tree (AST), control flow graph (CFG), and data flow graph (DFG) from the source code. Then, they are used to generate the code composite graph (CCG), which can be combined with a bidirectional graph neural network for vulnerability detection.

Source codes may have more intuitive information than assembly codes or executable codes. This information can better help detect vulnerabilities. For example, source codes can provide more syntactic and semantic information, which makes it easier to know how the data and inputs drive the paths of execution [25].

2.3. Vulnerability Detection by Assembly Code. Assembly codes are program files obtained by compiling the source codes. Practically, assembly codes might be easier to obtain in comparison with source code files. Grieco et al. [12] developed and implemented VDiscover. This system extracts different feature sets including static features of an assembly code and dynamic features during execution to solve large-scale vulnerability discovery. Xu et al. [21] presented a neural network-based cross-platform method to detect the similarity between assembly codes, which is an aid to detect vulnerabilities. Liu et al. [22] employed the attention mechanism on top of a bidirectional long short-term memory to detect assembly code vulnerabilities. Tian et al. [15] extracted code slices from assembly codes in their vulnerability detection system. A summary of the above studies is presented in Table 1, where we have highlighted the key differences of different works.

It has been found that vulnerability detection on the assembly code level can better solve cross-architecture problems [26]. Moreover, assembly codes are sometimes more sensitive to the semantic errors of a program [27]. This means that the assembly code can be used as a powerful supplement to improve the detector's performance.

Table 1: Reviewed studies for vulnerability detection.

	Data type	Network	Feature representation
SySeVR [10]	Source code	BGRU	SeVCs depicting semantic information induced by data dependency and control dependency
VulDeePecker [11]	Source code	BLSTM	Code gadget depicting data flow and control flow
BGNN4VD [20]	Source code	BGNN	CCG based on AST, CFG, and DFG
VDiscover [12]	Assembly code	Logistic regression, MLP of \single hidden layer, and random forest	Dynamic and static call sequences of library functions
Gemini [21]	Assembly code	Structure2vec	Attributed control flow graph (ACFG)
System [22]	Assembly code	Att-BiLSTM	Static call sequences of binary functions
BVDetector [15]	Assembly code	BGRU	Code slices depicting library/API function calls from binary programs

3. Designed System

3.1. System Overview. The proposed method analyzes the source code and assembly code of software to obtain more comprehensive information. First, we give the definitions of the source code slice and assembly code slice.

Definition 1 (source code slice). A source code slice. S_i is a snippet of a semantically related multiline source code, denoted as $S_i = (s_{i1}, s_{i2}, s_{i3}, \ldots, s_{in})$, where s_{ij}, $1 \le j \le n$, is the jth line in S_i.

Definition 2 (assembly code slice). An assembly code slice. D_i is a snippet of a semantically related multiline assembly code, denoted as $D_i = (d_{i1}, d_{i2}, d_{i3}, \ldots, d_{in})$, where d_{ij}, $1 \le j \le n$, is the jth line in D_i.

Figure 1 shows an example of the source code and corresponded assembly code, where assembly code D is compiled from source code S by using the GCC compiler. s_4 in S corresponds to (d_7, d_8) in D.

The workflow of our system is depicted in Figure 2. It has two phases: the training phase and the testing phase. In the training phase, the system first extracts source code slices from training codes and adds labels to them. Second, the system compiles training codes to assembly codes, from which assembly code slices are extracted and labeled. Third, source code slices and assembly code slices are aligned for further multimodal fusion. Fourth, the system converts source code slices and assembly code slices to vectors. Finally, the system trains a deep learning model by using these vectors. In the testing phase, the system tests the trained model using targeted codes and evaluates its performance.

3.2. Multimodal Network Structure. The designed system fuses assembly codes and source codes to predict the outcome. We use the multimodal hybrid fusion strategy [28] to conduct its network, as shown in Figure 3. It first combines assembly code slices and source code slices via early fusion. Then, these combined slices, together with the source code slices and assembly code slices, are fed into three individual networks. At last, the results of the three networks are used to give the final decision through late fusion.

3.2.1. Code Representation. We use code slices to represent different codes. Given a snippet of the source code, Joern [29] is first used to analyze it and generate a control flow graph (CFG), which represents the real-time execution of a process in the form of a graph. After that, we get the program dependency graph (PDG) from the source code, which is a graphical representation of the control dependency and data dependency among program statements. Finally, CFG and PDG are traversed forward and backward to extract all affected statements. These statements are combined to form a code slice, providing data for vulnerability detection. Figure 4 shows a running example of how we generate a code slice from CFG and PDG of a source code. All the affected statements of each function are captured through CFG and combined into a source code slice based on PDG.

Besides the assembly and source code slices, we further construct fused code slices by applying early fusion to them. We combine each source code slice S_i and the corresponded assembly code slice D_i to derive a new slice $G_i = (S_i, D_i)$. Moreover, redundant information in G_i caused by data fusion is removed, yielding a new set of code slices as part of the network input.

3.2.2. Network Model. The three types of slices are input into three individual networks having the same structure. Each of them consists of five layers: input layer, bidirectional LSTM (BLSTM) layer, dense layer, softmax layer, and output layer, as shown in Figure 5. The input layer is responsible for inputting vector data. The BLSTM layer extracts vulnerability characteristics from vulnerability samples. It contains 300 LSTM units in a bidirectional form (altogether 600 LSTM units). The dense layer reduces the dimensionality of the vector. We set up two dense layers to obtain better detection results. To prevent overfitting, we apply dropout with the value of 0.5 after the dense layers. The softmax layer represents and formats the classification results. At last, the output layer gives a decision. In our experiment, the loss

```
int main (int argc,char *argv[])
{
  float buf[10];

  buf[4105] = 55.55;

  return 0;
}
```

```
_main:
    push    %ebp
    movl    %esp, %ebp
    andl    $-16, %esp
    subl    $48, %esp
    call    __main
    movl    LC0, %eax
    movl    %eax, 16428(%esp)
    movl    $0, %eax
    leave
    ret
```

Figure 1: Demonstration of source code S and corresponded assembly code D.

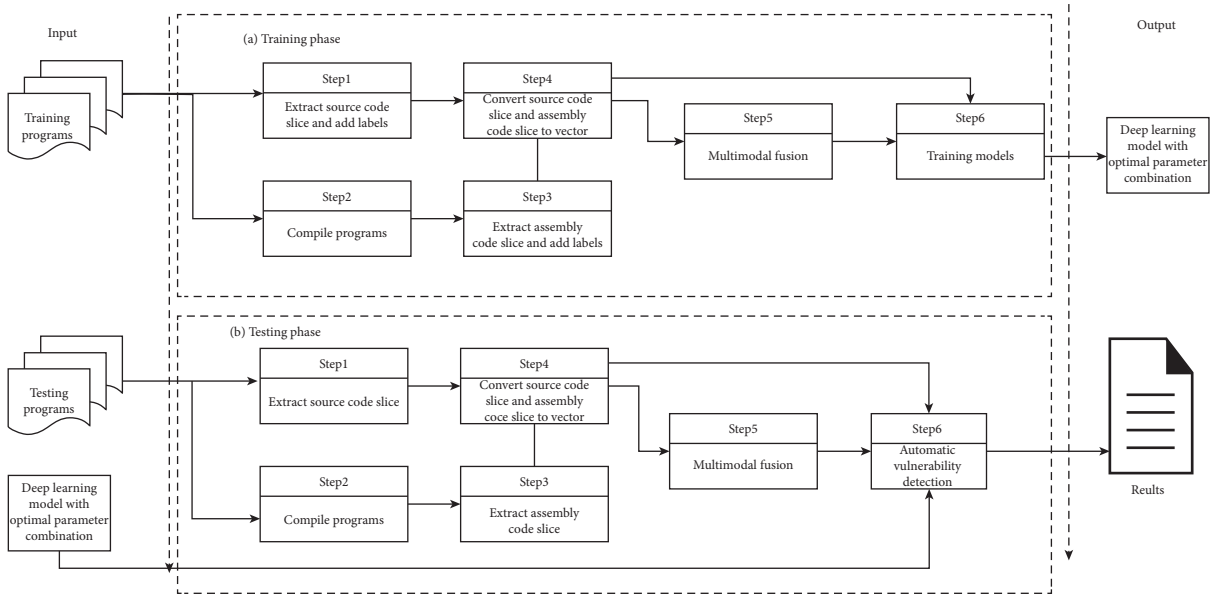


Figure 2: Workflow of the vulnerability detection system.

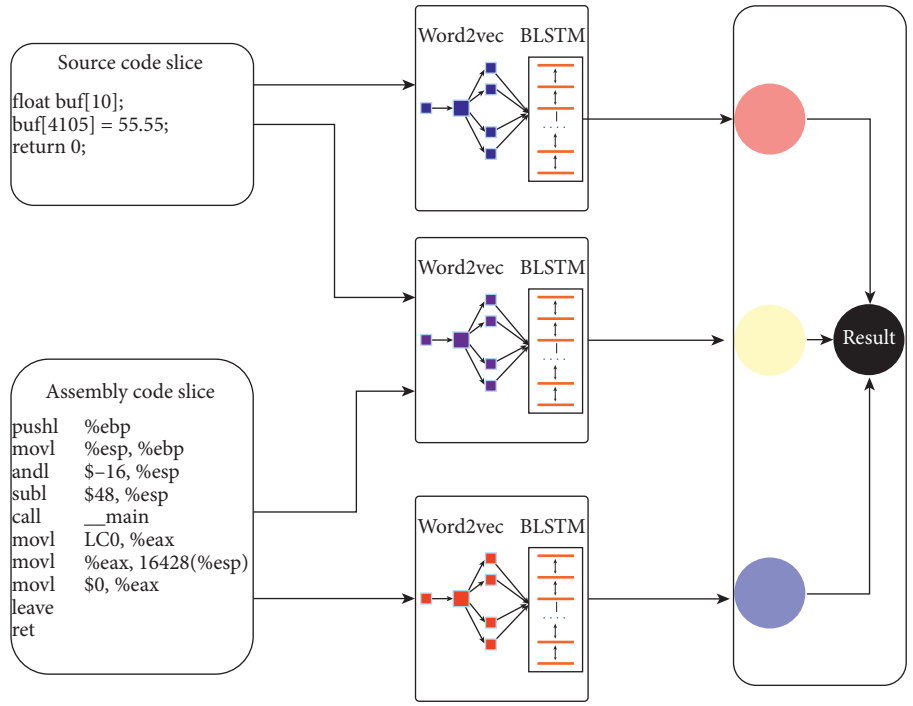


Figure 3: Network structure based on hybrid fusion.

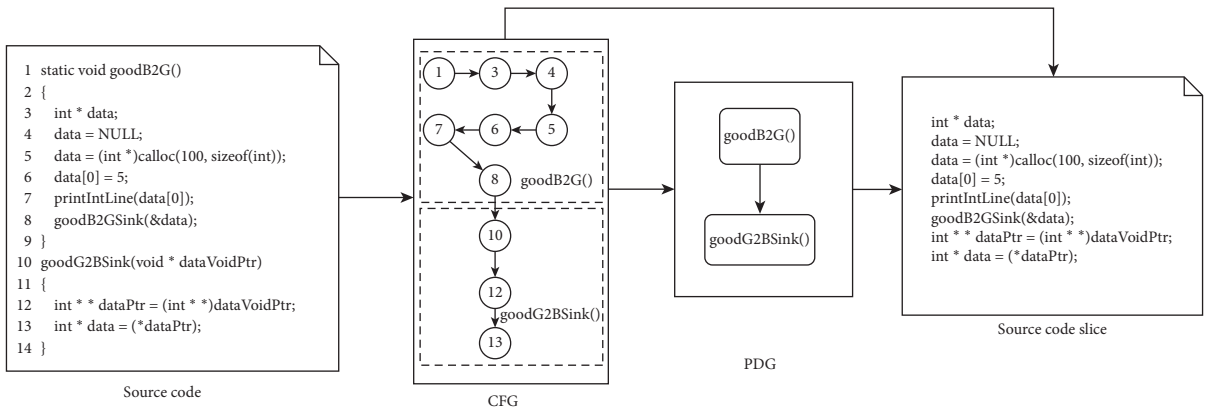


Figure 4: Example of extracting a source code slice.

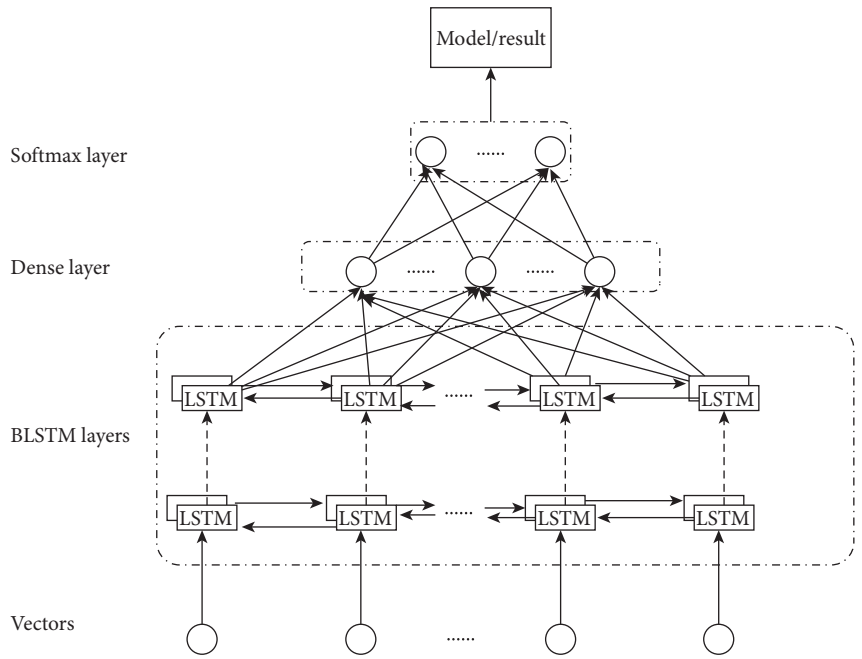


Figure 5: The structure of the individual network.

Table 2: Settings of the proposed network.

Layer name	Activation func.	Node num.
BLSTM	None	600
Dense 1	"LeakyReLU"	300
Dense 2	"LeakyReLU"	300
Softmax	"Softmax"	2

function used is categorical cross-entropy, and the optimizer used is Adamax [30]. During the training phase, we set the batch size with 64 and the epoch number with 30, which gives a good performance. More settings are detailed in Table 2.

3.2.3. Late Fusion Model. The decisions of the three networks may be different. As a result, at the end of our system, these decisions are passed through a voting layer to get the final result. Majority voting is employed in this layer. It is known that a late fusion model usually gets better results than those from a single model. It is expected to avoid erroneous decisions in individual models.

3.3. Training Phase. The training phase is highlighted in Figure 2. It has 5 steps:

Step 1. Source Code Processing.

(1) Extract source code slices: extract source code slices from a training code according to some vulnerability syntax characteristics.

(2) Add labels: give each source code slice a label as the indicator of vulnerabilities ("1" means the presence of vulnerabilities, and "0" means the absence of vulnerability). The method on how to determine the vulnerability of a code slice is detailed in Section 3.6.

Step 2. Program Compiling

(1) Generate an assembly code from the training code using the GCC compiler.

Step 3. Assembly Code Processing

(1) Extract assembly code slices: use the code slice alignment algorithm detailed in Section 3.5 to analyze the assembly code, and extract assembly code slices corresponding to source code slices.

(2) Add labels: similar to the processing of source code slices, each assembly code slice is assigned a label indicating whether it contains vulnerabilities ("1" means yes and "0" means no).

Step 4. Convert Source Code Slices and Assembly Code Slices into Vectors

(1) As shown in Figure 6, the system uses word2vec to convert source code slices, assembly code slices, and hybrid code slices into vector forms, which will be input into the deep learning model.

Step 5. Model Training

(1) Use the data generated from Steps 1–4 to train the network presented in Section 3.2. The parameters and design of the model have been detailed in Section 3.2.

3.4. Testing Phase. As highlighted in Figure 2, the testing phase has 5 steps.

Steps 1–4. Similar to Steps 1–4 in the training phase, except that the label adding is in no need.

Step 5. Automatic Vulnerability Detection.

(1) Input the data generated from Step 1 to Step 4 into the trained model, and the system will give a detection result. Result "1" means that vulnerabilities exist in this code slice, and "0" means there is no vulnerability.

3.5. Code Alignment. In this section, we provide the algorithm employed in data alignment, align_data. The top level of the algorithm is listed in Algorithm 1.

The whole algorithm can be summarized into three stages, namely, (1) pseudocode generation, (2) collection of candidate sets of matched assembly codes, and (3) best match finding. At stage (1), we use IDA Pro [31] to generate pseudocodes, pe_code, from the assembly code (i.e., Line 3 in Algorithm 1), where each statement in pe_code corresponds to several statements $d_{iu}, \ldots, d_{iw}$ in assembly_code. At stage (2), we search for the candidate set of assembly code statements that match s_{ij} (i.e., Lines 4–9 in Algorithm 1). For each p_i, if its statement type (e.g., loop statement and assignment statement) is as same as s_{ij}., the corresponded $d_{iu}, \ldots, d_{iw}$ can be considered as candidate matches of s_{ij}. At stage (3), the Hungarian algorithm [32] is used to get a slice $(d_{i1}, \ldots, d_{ij})$ from the candidate set D_i'. It is considered as a potential match for S_i and combined into D_i. Then, we use string and integer constants, function and library calls, and function declaration information to compute the similarity between D_i and S_i. If this similarity is bigger than a threshold, a satisfactory match is achieved. Otherwise, repeat this stage until a satisfactory match is found (i.e., Lines 11–16 in Algorithm 1).

3.6. Slice Labeling. A simple tool is developed here to give each code slice a label indicating the presence of vulnerabilities. All the vulnerability functions have been defined and given by the NIST Software Assurance Reference Dataset (SARD) [33]. This paper manually populates these vulnerability functions according to their vulnerability codes provided by SARD. The procedure of the tool is described as follows:

(1) Obtain the file names, vulnerability locations, vulnerability types, and other information from the vulnerability file.

(2) Traverse the training codes and the vulnerability information to obtain the vulnerability code and vulnerability function names. Collect this information to populate the vulnerability function library.

(3) Traverse each code slice and vulnerability function library to determine whether the code slice has vulnerability functions or call vulnerability functions.

(4) Divide each code slice into two categories:

   (1) If a code slice has vulnerability functions or call vulnerability functions, it will be labeled with "1."
   (2) Otherwise, it will be labeled with "0."

4. Experimental Results

Our experiments are conducted on a computer with an NVIDIA GeForce GTX 2080 Ti GPU and an Intel Xeon $E5-2678$ v3 CPU operating at 2.50 GHz. The employed program compiler is GCC 9.3.0.

4.1. Evaluation Metrics. In this paper, we use five standard indicators suggested in [34] to measure the performance of the vulnerability detection system. We denote TP as the number of vulnerable samples that are detected as vulnerable (i.e., true-positives), FP as the number of samples that are not vulnerable but are detected as vulnerable (i.e., false-positives), TN as the number of samples that are not vulnerable and are not detected as vulnerable (i.e., true-negatives), and FN as the number of vulnerable samples that are not detected as vulnerable (i.e., false-negatives). Then, the five indicators, namely, accuracy (A), false positive rate (FPR), false negative rate (FNR), precision (P), and F1-measure (F1), are defined as

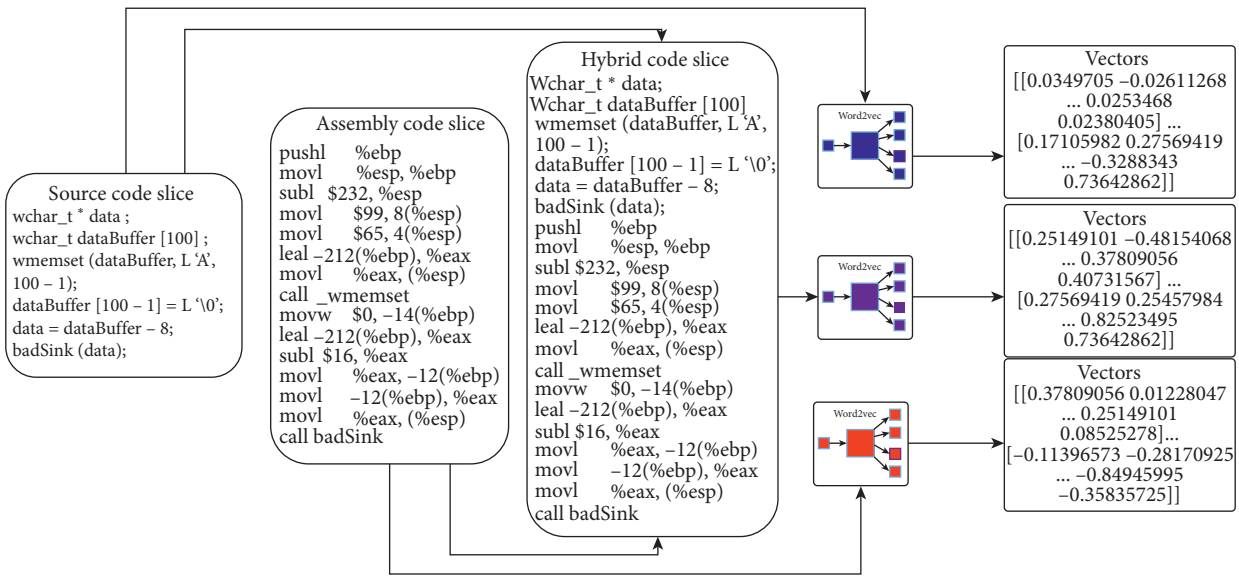


Figure 6: Convert code slices into vectors.

(1) **function** ALIGN_DATA(S_i, assembly_code)
(2) **for** s_{ik} in S_i **do**
(3) pe_code ⟵ IDA_PRO (assembly_code)//p_i in pe_code corresponds to $d_{iu}, \ldots, d_{iw}$ in assembly_code
(4) $D_i' \longleftarrow \varnothing$
(5) **for** p_i in pe_code **do**
(6) **if** STATEMENT_TYPE (p_i) ==STATEMENT_TYPE s_{ik} **then**
(7) $D_i' \longleftarrow D_i' \cup (d_{iu}, \ldots, d_{iw})$
(8) **end if**
(9) **end for**
(10) **end for**
(11) similarity_score ⟵ 0
(12) **while** similarity_score < GOAL **do**
(13) $(d_{i1}, \ldots, d_{ij}) \longleftarrow$ HUNGARIAN $((s_{i1}, \ldots, s_{ij}), D_i')$
(14) $D_i \longleftarrow$ GEN_ SLICE $((d_{i1}, \ldots, d_{ij}))$
(15) similarity_score ⟵ SIMILARITY (S_i, D_i)
(16) **end while**
(17) **return** D_i
(18) **end function**

Algorithm 1: Align_data algorithm.

$$FPR = \frac{\text{FP}}{\text{FP} + \text{TN}},$$

$$FNR = \frac{\text{FN}}{\text{TP} + \text{FN}},$$

$$A = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \quad (1)$$

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

$$F1 = \frac{2 \cdot P \cdot (1 - \text{FNR})}{P + (1 - \text{FNR})}.$$

4.2. Dataset Preparing

4.2.1. Input Preparing. In this paper, we collect a large number of source codes from the NIST Software Assurance Reference Dataset (SARD) [33]. Then, these codes are compiled with GCC on Windows X64 to obtain the corresponded assembly codes. We randomly select 80% of the source codes and corresponded assembly codes as the training dataset, and the remaining 20% are used for the testing.

To granularly evaluate the proposed system, we divide the test cases into the following four categories:

(1) Library/API function call (FC), namely, syntax vulnerability characteristics related to library/API function calls

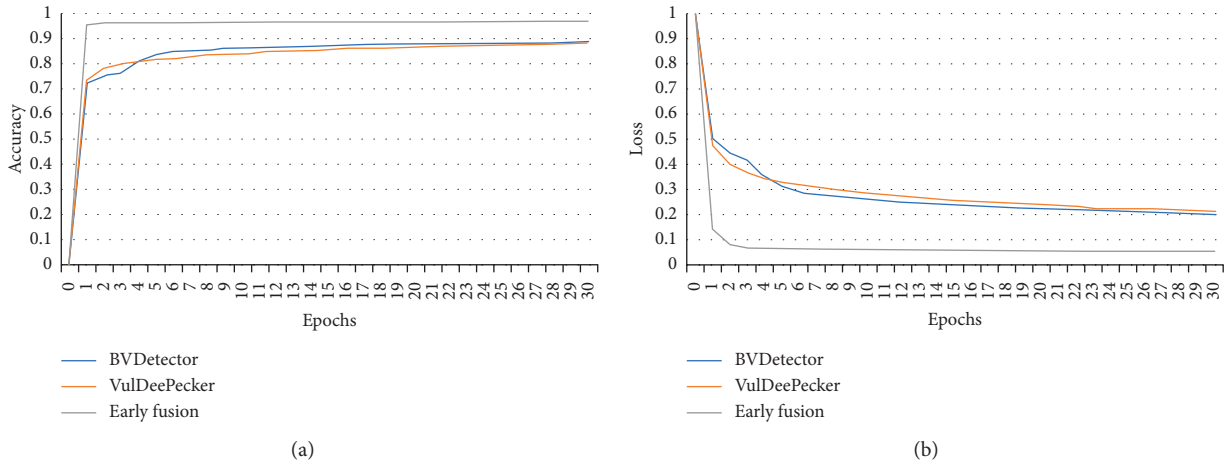


Figure 7: The training accuracy curves (a) and training loss curves (b) of compared systems.

(2) Array usage (AU), namely, vulnerability characteristics related to array element access

(3) Pointer usage (PU), namely, syntax vulnerability characteristics related to pointer usage

(4) Arithmetic expression (AE), namely, syntax vulnerability characteristics related to improper arithmetic expressions

4.2.2. Extracting Code Slices and Adding Labels

(1) Extracting Code Slices. Source code slices are extracted first. We obtain CFG and PDG by parsing the source code. After that, all the statements affected by the library/API function calls are extracted based on CFG and PDG. These statements form source code slices. Then, we use Algorithm 1 to collect assembly code slices corresponding to the source code. In total, 42938 source code slices and assembly code slices are extracted.

(2) Add Labels. We obtain file names, vulnerability locations, vulnerability types, and other information by analyzing program documents in the SARD project. Then, they are used to build a vulnerability function database, where all the code slices are labeled by the tool developed in Section 3.6. In total, we label 16478 code slices with "1," and 26460 code slices with "0."

To ensure effective training of the detection system, 16478 code slices are randomly selected from the 26460 code slices labeled with "0." They are then combined with all the code slices labeled with "1" to form a dataset in each training/testing round, of which 13182 code slices labeled with "0" and 13182 code slices labeled with "1" are randomly selected to train the model, and the rest are used for testing.

4.2.3. Convert Datasets into Vectors. The neural network only accepts vectors, thus we need to convert the data into the vector form, which is carried out by word2vec. Since the effectiveness of our system depends largely on the quality of the word embeddings produced, we first use the constructed dataset as a corpus to train word2vec. It can give a trained word2vec model more suitable for our task. After that, code slices are divided into a series of tokens. Each token is converted through the trained word2vec to obtain a fixed-length vector that can be recognized by the network. In this experiment, the length of each code slice is 50 and the dimension of the word embedding is 100.

4.3. Training and Results

4.3.1. Network Training. In this experiment, we use Keras with TensorFlow to implement the system. The length of the input layer is fixed to be 200, and the number of nodes of each BLSTM is 300. Two fully connected layers with LeakyReLU are employed as its activation functions. The number of nodes is 300, and the dropout is 0.5. The model is optimized by Adamax [30] with a learning rate of 0.002.

4.3.2. Result. We compare the proposed system with the approaches presented in [35] (denoted as VulDeePecker) and [15] (denoted as BVDetector). VulDeePecke detects vulnerabilities at the source code level, while BVDetector detects vulnerabilities at the assembly code level. The training accuracy and training loss curves for these systems are compared in Figure 7. It can be observed that all the compared systems converge fast. The average accuracy of compared systems is close to 90%, while the accuracy of our early fusion system can reach 97%.

Table 3 lists the number of marked code slices by different schemes. It can be observed that some vulnerabilities can be detected by the model based on source codes, but not by the model based on assembly codes. In contrast, some vulnerabilities can only be detected by the model based on assembly codes. Take the code fragment shown in Figure 8 as an example, which is a vulnerability related to out-of-bounds array access. The system, which uses source code slices, may incorrectly detect the

Table 3: Comparison of different metrics on the number of marked code slices.

Metrics	VulDeePecker [35]	BVDetector [15]	Early fusion	Hybrid fusion
Original number of code slices		6591		
Number of code slices marked as safe	2977	3102	3156	3202
Number of code slices marked as vulnerable	3614	3489	3435	3389
Number of correctly marked code slices	5665	5829	6373	6421
Number of incorrectly marked code slices	926	762	218	170

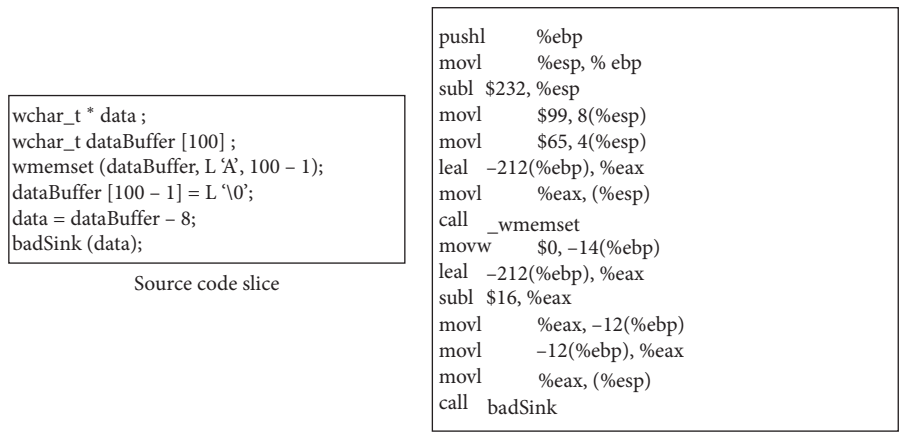


Figure 8: Example of a vulnerability in a source code slice and the corresponded assembly code slice.

Table 4: Test results for different categories.

Kind		FPR (%)	FNR (%)	A (%)	P (%)	F1 (%)
FC	VulDeePecker [35]	9.6	17.5	85.9	91.3	86.6
	BVDetector [15]	7.7	13.9	88.9	92.8	89.3
	Early fusion	**1.1**	5.0	96.8	**98.9**	96.8
	Hybrid fusion	1.3	**4.5**	**97.0**	98.7	**97.0**
AU	VulDeePecker [35]	9.4	17.4	86.1	91.5	86.8
	BVDetector [15]	7.4	13.8	89.1	93.1	89.4
	Early fusion	**1.1**	5.1	96.7	**98.9**	96.8
	Hybrid fusion	1.3	**4.5**	**96.9**	98.7	**97.0**
AE	VulDeePecker [35]	9.7	17.5	85.9	91.2	86.6
	BVDetector [15]	7.2	15.4	88.3	93.0	88.5
	Early fusion	**1.0**	5.9	96.4	**98.9**	96.4
	Hybrid fusion	1.4	**5.0**	**96.6**	98.4	**96.6**
PU	VulDeePecker [35]	6.7	18.1	86.9	93.8	87.4
	BVDetector [15]	7.8	13.9	88.8	92.7	89.2
	Early fusion	**1.1**	5.0	96.8	**98.9**	96.8
	Hybrid fusion	1.3	**4.5**	**97.0**	98.6	**97.0**

vulnerability as normal pointer arithmetic because it is difficult to detect out-of-bounds array access in a calculation. By contrast, the system using assembly code slices can easily detect this vulnerability via memory addresses. As a result, fusing source codes and assembly codes in our model can accurately detect a wider range of vulnerabilities.

The comparison results on the test set are shown in Tables 4 and 5, where a representative static vulnerability mining tool on the market, Flawfinder [36], is also included for comparison. It indicates that our system achieves better results in all aspects compared to other systems. Compared with VulDeePecke and BVDetector, our hybrid fusion-based system can raise the F1 scores by 10%. The score of our

Table 5: Test results for different systems.

	Kind	FPR (%)	FNR (%)	*A* (%)	*P* (%)	F1 (%)
Flawfinder [36]	ALL	10.2	81.3	60.6	48.3	24.6
VulDeePecker [35]	ALL	9.7	17.4	85.9	91.3	86.6
BVDetector [15]	ALL	7.7	13.9	88.9	92.8	89.3
Early fusion	ALL	**1.1**	5.0	96.8	**98.9**	96.8
Hybrid fusion	ALL	1.3	**4.5**	**96.9**	98.6	**97.0**

system is also well above the one obtained by Flawfinder. This confirms the effectiveness of the proposed vulnerability detection system.

5. Conclusion

This paper proposes a system for detecting vulnerabilities in software through deep learning. It combines the vulnerability features at the source code level and assembly code level to improve the ability of vulnerability detection. The system extracts code slices from both the source code and the assembly code of a program. Then, a code alignment algorithm based on string, integer constants, function and library calls, etc., is suggested to align these code slices. The hyper fusion-based deep learning model considers early fusion and late fusion. Early fusion combines aligned source code slices and assembly code slices to generate a new dataset, while late fusion combines the decisions of the deep learning models on the source dataset, assembly dataset, and fused dataset. We implement a prototype and perform systematic experiments. The experimental results show the effectiveness of the proposed system. In future research, we can extend this method to cross languages and cross platforms. Moreover, we could consider additional data besides source codes and assembly codes.

Data Availability

The source code and dataset have been deposited in the GitHub repository (https://github.com/onstar99/VulnerabilitySystem).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Key R&D Program of Guangdong Province (Grant no. 2019B010136003), National Natural Science Foundation of China (Grant nos. 61802145 and 61932010), Natural Science Foundation of Guangdong Province, China (Grant no. 2019B010137005), Science and Technology Program of Guangzhou, China (Grant no. 202007040004), Fundamental Research Funds for the Central Universities, Opening Project of State Key Laboratory of Information Security, and Opening Project of Guangdong Key Laboratory of Intelligent Information Processing and Shenzhen Key Laboratory of Media Security.

References

[1] CVE, https://cve.mitre.org/.

[2] E. Stepanov and K. Serebryany, "Memorysanitizer: fast detector of uninitialized memory use in c++," in *Proceedings of the 2015 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pp. 46–55, San Francisco, CA, USA, March 2015.

[3] M. Vimpari, "An evaluation of free fuzzing tools," Master's thesis, University of Oulu, Oulu, Finland.

[4] S. M. Ghaffarian and H. R. Shahriari, "Software vulnerability analysis and discovery using machine-learning and data-mining techniques," *ACM Computing Surveys*, vol. 50, no. 4, pp. 1–36, 2017.

[5] S. Kim, S. Woo, H. Lee, and H. Oh, "Vuddy: a scalable approach for vulnerable code clone discovery," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, pp. 595–614, San Jose, CA, USA, June 2017.

[6] J. Jang, A. Agrawal, and D. Brumley, "Redebug: finding unpatched code clones in entire os distributions," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, pp. 48–62, San Jose, CA, USA, July 2012.

[7] N. H. Pham, T. T. Nguyen, H. A. Nguyen, and T. N. Nguyen, "Detection of recurring software vulnerabilities," in *Proceedings of the IEEE/ACM International Conference on Automated Software Engineering*, pp. 447–456, Antwerp, Belgium, September 2010.

[8] A. Hovsepyan, R. Scandariato, W. Joosen, and J. Walden, "Software vulnerability prediction using text analysis techniques," in *Proceedings of the 4th International Workshop on Security Measurements and Metrics*, pp. 7–10, Lund, Sweden, September 2012.

[9] Y. Pang, X. Xue, and A. S. Namin, "Predicting vulnerable software components through n-gram analysis and statistical feature selection," in *Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications*, pp. 543–548, Miami, FL, USA, December 2015.

[10] Z. Li, D. Zou, S. Xu, H. Jin, Y. Zhu, and Z. Chen, "Sysevr: a framework for using deep learning to detect software vulnerabilities," *IEEE Transactions on Dependable and Secure Computing*, vol. 2021, Article ID 3051525, 1 page, 2021.

[11] Z. Li, D. Zou, S. Xu, Z. Chen, Y. Zhu, and H. Jin, "Vuldeelocator: a deep learning-based fine-grained vulnerability detector," arXiv preprint arXiv:2001.02350.

[12] G. Grieco, G. L. Grinblat, L. Uzal, S. Rawat, J. Feist, and L. Mounier, "Toward large-scale vulnerability discovery using machine learning," in *Proceedings of the 6th ACM Conference on Data and Application Security and Privacy*, pp. 85–96, New Orleans, LA, USA, March 2016.

[13] A. Younis, Y. Malaiya, C. Anderson, and I. Ray, "To fear or not to fear that is the question: code characteristics of a vulnerable functionwith an existing exploit," in *Proceedings of the 6th ACM Conference on Data and Application Security and Privacy*, pp. 97–104, New Orleans, LA, USA, March 2016.

[14] Q. Feng, R. Zhou, C. Xu, Y. Cheng, B. Testa, and H. Yin, "Scalable graph-based bug search for firmware images," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 480–491, Vienna, Austria, October 2016.

[15] J. Tian, W. Xing, and Z. Li, "Bvdetector: a program slice-based binary code vulnerability intelligent detection system," *Information and Software Technology*, vol. 123, Article ID 106289, 2020.

[16] G. Papandreou, A. Katsamanis, V. Pitsikalis, and P. Maragos, "Multimodal fusion and learning with uncertain features applied to audiovisual speech recognition," in *Proceedings of the 9th Workshop on Multimedia Signal Processing*, pp. 264–267, Chania, Crete, Greece, October 2007.

[17] S. Bedoya and T. H. Falk, "Laughter detection based on the fusion of local binary patterns, spectral and prosodic features," in *Proceedings of the 18th International Workshop on Multimedia Signal Processing*, pp. 1–5, Montreal, QC, Canada, September 2016.

[18] G. A. Ramirez, T. Baltrušaitis, and L.-P. Morency, "Modeling latent discriminative dynamic of multi-dimensional affective signals," in *Proceedings of the International Conference on Affective Computing and Intelligent Interaction*, pp. 396–406, Memphis, TN, USA, October 2011.

[19] G. Castellano, L. Kessous, and G. Caridakis, "Emotion recognition through multiple modalities: face, body gesture, speech," in *Affect and Emotion in Human-Computer Interaction*, pp. 92–103, Springer, Berlin, Heidelberg, 2008.

[20] S. Cao, X. Sun, L. Bo, Y. Wei, and B. Li, "Bgnn4vd: constructing bidirectional graph neural-network for vulnerability detection," *Information and Software Technology*, vol. 136, Article ID 106576, 2021.

[21] X. Xu, C. Liu, Q. Feng, H. Yin, L. Song, and D. Song, "Neural network-based graph embedding for cross-platform binary code similarity detection," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 363–376, Dallas, TX, USA, October 2017.

[22] S. Liu, M. Dibaei, Y. Tai, C. Chen, J. Zhang, and Y. Xiang, "Cyber vulnerability intelligence for internet of things binary," *IEEE Transactions on Industrial*, vol. 16, no. 3, pp. 2154–2163, 2019.

[23] D. Zou, S. Wang, S. Xu, Z. Li, and H. Jin, "μVulDeePecker: a deep learning-based system for multiclass vulnerability detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 2019, Article ID 2942930, 1 page, 2019.

[24] word2vec, 2018, https://radimrehurek.com/gensim/models/word2vec.html.

[25] T. Le, T. Nguyen, T. Le et al., "Maximal divergence sequential autoencoder for binary software vulnerability detection," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, May 2018.

[26] F. Zuo, X. Li, P. Young, L. Luo, Q. Zeng, and Z. Zhang, "Neural machine translation inspired binary code similarity comparison beyond function pairs," arXiv preprint arXiv: 1808.04706.

[27] A. V. Phan and M. Le Nguyen, "Convolutional neural networks on assembly code for predicting software defects," in *Proceedings of the 2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems*, pp. 37–42, Hanoi, Vietnam, November 2017.

[28] Z.-Z. Lan, L. Bao, S.-I. Yu, W. Liu, and A. G. Hauptmann, "Multimedia classification and event detection using double fusion," *Multimedia Tools and Applications*, vol. 71, no. 1, pp. 333–347, 2014.

[29] F. Yamaguchi, N. Golde, D. Arp, and K. Rieck, "Modeling and discovering vulnerabilities with code property graphs," in *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, pp. 590–604, Washington, DC, USA, May 2014.

[30] D. P. Kingma, J. Ba, and Adam, "Adam: a method for stochastic optimization," arXiv preprint arXiv:1412.6980.

[31] "IDA pro multi-processor disassembler and debugger," http://www.hexrays.com/products/ida/index.shtml.

[32] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[33] "Software assurance reference dataset," 2018, https://samate.nist.gov/SRD/index.php.

[34] M. Pendleton, R. Garcia-Lebron, J.-H. Cho, and S. Xu, "A survey on systems security metrics," *ACM Computing Surveys*, vol. 49, no. 4, pp. 1–35, 2016.

[35] Z. Li, D. Zou, S. Xu et al., "Vuldeepecker: a deep learning-based system for vulnerability detection," arXiv preprint arXiv:1801.01681.

[36] Flawfinder: http://www.dwheeler.com/flawfinder.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6689486, 12 pages
https://doi.org/10.1155/2021/6689486

Research Article

Permission Sensitivity-Based Malicious Application Detection for Android

Yubo Song[1,2] **Yijin Geng**[1,2] **Junbo Wang**[3,4] **Shang Gao**[5] **and Wei Shi**[1,2]

[1]*Key Laboratory of Computer Networking Technology of Jiangsu Province, School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China*
[2]*Purple Mountain Laboratories, Nanjing 211189, China*
[3]*School of Information Science and Engineering, Southeast University, Nanjing 211189, China*
[4]*National Mobile Communications Research Laboratory, Nanjing 211189, China*
[5]*Computing Department, The Hong Kong Polytheistic University, Hung Hom, Hong Kong*

Correspondence should be addressed to Yubo Song; songyubo@seu.edu.cn

Received 29 December 2020; Accepted 20 July 2021; Published 29 July 2021

Academic Editor: Liguo Zhang

Since a growing number of malicious applications attempt to steal users' private data by illegally invoking permissions, application stores have carried out many malware detection methods based on application permissions. However, most of them ignore specific permission combinations and application categories that affect the detection accuracy. The features they extracted are neither representative enough to distinguish benign and malicious applications. For these problems, an Android malware detection method based on permission sensitivity is proposed. First, for each kind of application categories, the permission features and permission combination features are extracted. The sensitive permission feature set corresponding to each category label is then obtained by the feature selection method based on permission sensitivity. In the following step, the permission call situation of the application to be detected is compared with the sensitive permission feature set, and the weight allocation method is used to quantify this information into numerical features. In the proposed method of malicious application detection, three machine-learning algorithms are selected to construct the classifier model and optimize the parameters. Compared with traditional methods, the proposed method consumed 60.94% less time while still achieving high accuracy of up to 92.17%.

1. Introduction

Nowadays, Android operating system occupies 85% of the market share and has become the most popular mobile operating system. However, owing to the increase in usage and the source's openness, the Android system becomes the target of malware. According to the report on China's mobile phone security status in the first half of 2020, 360 Security Brain intercepted approximately 1.048 million new malware samples on the mobile terminal, approximately 6 thousand new ones added every day. These malicious applications hidden in some application stores have access to users' privacy, resulting in information leakage of private data and economic loss, affecting the Android ecosystem's healthy development [1, 2]. Therefore, the research on efficient malware detection methods for application stores is very crucial. However, the traditional detection methods based on permission features mostly have the following problems:

(i) The legality of requesting the same permission is different in various categories of applications. For example, requesting permission to read contacts is a legal behavior in social chat applications, but it is usually regarded as a malicious behavior in photo-taking applications. In general, putting all types of Android applications in the same data set together may cause misjudgment.

(ii) Many malicious behaviors need to call multiple permission combinations. For example, the

malicious behavior of uploading contact information to a website requires calling READ_CONTACTS and Internet permissions. Therefore, analyzing the information of permission invocation, ignoring the influence of some specific permission combination on the malware detection will result in the decrease of the detection accuracy.

(iii) Some permission features cannot effectively distinguish between benign applications and malicious applications. Taking such permission features into consideration will lead to many invalid features, which will reduce the accuracy of detection and cost more detection time.

To solve the above problems, considering the coarse-grained authorization mechanism of the Android system, we present an Android malware detection method based on permission sensitivity. This method screens the permission features and permission combination features based on permission sensitivity to obtain the sensitive permission feature sets of various applications. The weight allocation method is then used to quantify the sensitive permission features, thus realizing malware detection. The main contributions of our work are summarized as follows:

(i) An automatic classification method based on semantic analysis is proposed. The application's primary function is identified through the semantic analysis of the application description text, and then the label category can be attached to the application.

(ii) Considering the sensitivity of permission features and permission combination features under various category labels are different, the permission feature set is filtered to obtain the application's sensitive permission feature set to perform malicious application detection.

(iii) We propose a method of processing permission features based on weight allocation. Considering that permission features have different abilities to distinguish malicious and benign applications, the corresponding weights are assigned to different features. Thus application permission request information can be converted into digital characteristics for malicious application detection.

(iv) The experimental results show that the accuracy rate of the semantic analysis-based classification method is 91.65%. The accuracy rate of the malware detection method is 92.17%, and this method only requires 0.1256 seconds to detect a single sample on average, which is 60.94% less than the traditional detection method.

2. Related Works

The classification methods of Android applications are mainly divided into two types: manual classification and automatic classification. The manual classification method determines the application's basic functions by the application store staff through the application's trial run; thereby, the application's label can be determined. This method's operation complexity is low, but the time cost and labor costs are enormous, and the error rate is inevitable.

Automatic classification is the process that the classifier model reads and analyzes the code file or other useful information of the application to determine the application's category according to the predefined category label set. Compared with the manual classification method, the automatic classification method, with the help of powerful computing and storage capacity of the computer, eliminates manual trial operation and simulated interactive operation, which dramatically improves application category determination efficiency. The application category automatic determination method is mainly aimed to find the key features in the static code files of Android applications for processing and analysis, which is also called the static analysis method. Burguera et al. [3] parsed the Android application software package to extract static features, combined them with the APK folder's size, thus can perform feature filtering and fusion to form joint features for classification. After that, the joint features are formed for classification. Yuan et al. [4] parsed and decompiled the APK file, extracted the API call information and the string as features for classification. However, decompiling the APK file of the software package to extract features for classification requires a long work cycle, which affects the classification efficiency. Kutlay and Karaduzovic-Hadziabdic [5] and Ahmad et al. [6] both used static analysis to extract features and construct classifiers for classification through machine-learning algorithms. However, these methods usually use an unsupervised learning clustering algorithm to construct classifiers, dividing the static features into several clusters, not determining the specific category label in the Android App store.

Several methods have been proposed for Android malware detection, including dynamic analysis, static analysis, and hybrid analysis. Each method has its merits and shortcomings.

Dynamic analysis refers to the methods that deploy the application on the emulator or actual device to execute it entirely. Simulating the interaction between users and applications can determine whether there is any malicious behavior [7, 8]. Wu et al. [7] used the system call frequency and system call dependence to construct joint features for malicious detection, and the accuracy rate was 93%. Jeon et al. [8] performed dynamic analysis on malware to extract related malicious behaviors such as system calls. They used them as a basis for discrimination to compare the application's behavior data to be tested, determining the application's malicious degree. Although these malware detection methods based on dynamic analysis have high accuracy, they cannot be widely used because of the difficulty of execution and high cost of time.

The static analysis method realizes detection by analyzing the application's static code and does not need to run the application entirely. Because of omitting the running process, the static analysis method consumes fewer hardware resources and time, so it is a relatively efficient and fast detection method. Lindorfer et al. [9] combined the

permission request information of the application with the sensitive API call information to form a joint feature for malicious detection. Guen et al. [10] extracted seven functions from the APK file by analyzing the list file and DEX file. These functions enrich the extracted information to express the application's characteristics to investigate whether the application under test has malicious behavior. However, these methods ignore the influence of application category on the accuracy, that is, the legality of the same permission request information is different in different types of applications. Besides, static analysis methods are hard to defend against obfuscation and repackaging.

Hybrid analysis refers to the combination of static analysis and dynamic analysis for malicious application detection. Therefore, hybrid analysis has the advantages of both. This method analyzes the APK folder of the Android application package and various application runtime behaviors, which is considered a comprehensive analysis method. However, the hybrid analysis also inherits the apparent disadvantages of dynamic analysis, which will lead to excessive consumption of Android system resources and a large amount of time waste [11].

In summary, dynamic analysis, static analysis, and hybrid analysis all have certain drawbacks, and they are not suitable for large-scale malware detection in the Android application market.

3. Detection Methodology

The process of android malware detection based on permission sensitivity is illustrated in Figure 1, and the details are provided as follows:

(i) Data set preprocessing: The data set preprocessing includes two parts. One is to obtain the application category label; the other is to obtain the application permission invocation information. The category label of the application is determined by the semantic analysis based on the application description text. The permission call information is gathered by decompiling the APK file of the Android application installation package and parsing the relevant files.

(ii) Permission feature processing: Permission feature processing consists of two parts, the extraction of sensitive permission feature set and the feature quantification based on the weight allocation method, which is convenient for inputting the classifier model training. The acquisition of the sensitive permission feature set is obtained by filtering permission features and permission combination features based on permission sensitivity. The feature quantification is based on the weight allocation method, which quantifies the application sensitive permission request information into the digital features to measure the application's malicious degree.

(iii) Classifier model construction: The classifier model is constructed based on the appropriate machine-learning algorithm. The malicious detection of the application under test is completed by training sensitive digital features.

3.1. Permission Sensitivity. The permission features and permission combination features are filtered based on the permission sensitivity in detecting malicious applications. A representative set of sensitive permission features is obtained for each type of application. Besides, to accurately represent the distinguishing strength of different permissions on malicious applications, the feature set is divided into two categories corresponding to the permission sensitivity, namely, lost permission and overload permission. In this way, different weights are assigned to the two types of permissions based on the weight allocation method, realizing the quantification of sensitive permissions. For the permission set $P_{\mathrm{er}} = P_{\mathrm{er1}}, P_{\mathrm{er2}}, \ldots, P_{\mathrm{er}k}$, there are the following definitions:

3.1.1. Permission Matrix. For the permission call information of a single application sample under the category label, the permission vector is obtained by comparing it with the permission set:

$$\overrightarrow{\mathrm{PV}_{t_i j}} = \left(\mathrm{pv}_1, \mathrm{pv}_2, \ldots, \mathrm{pv}_k\right)^T. \tag{1}$$

Each position in the vector can only take 0 or 1. The value of 0 means that the application does not call the permission of the corresponding position in the permission set. Otherwise, the value is 1. The permission matrix is obtained by multiplying the permission vector with its transposition matrix:

$$\mathrm{PM}_{t_i j} = \overrightarrow{\mathrm{PV}_{t_i j}} \times \overrightarrow{\mathrm{PV}_{t_i j}}^T = \left[\mathrm{pm}_{mn}\right] 0 \le m \le k, \quad 0 \le n \le k. \tag{2}$$

The value of each position in the permission matrix is 0 or 1. pm_{ii} $(0 \le i \le k)$ on the diagonal line of the matrix represents the request status of the permission feature with the sample application. The value of 1 means that the application has called the ith position permission in the set P_{er}, otherwise, the value is 0. pm_{ij} $(0 \le i \le k, 0 \le j \le k, i \ne j)$ on the nondiagonal position of the matrix indicates the invocation of the combination of the ith permission and the jth permission feature in the sample application. A value of 0 means that the application does not call the permission combination, and a value of 1 means that the sample application calls the permission combination.

3.1.2. Average Permission Matrix. The average permission matrix of benign applications under t_i category label is obtained by summing the permission matrix of all benign applications under t_i category label and dividing by the total number of benign applications:

$$\overline{\mathrm{PM}_{t_i B}} = \frac{\sum_{j=1}^{\mathrm{size}\left(t_i B\right)} \mathrm{PM}_{t_i B_j}}{\mathrm{size}\left(t_i B\right)}. \tag{3}$$

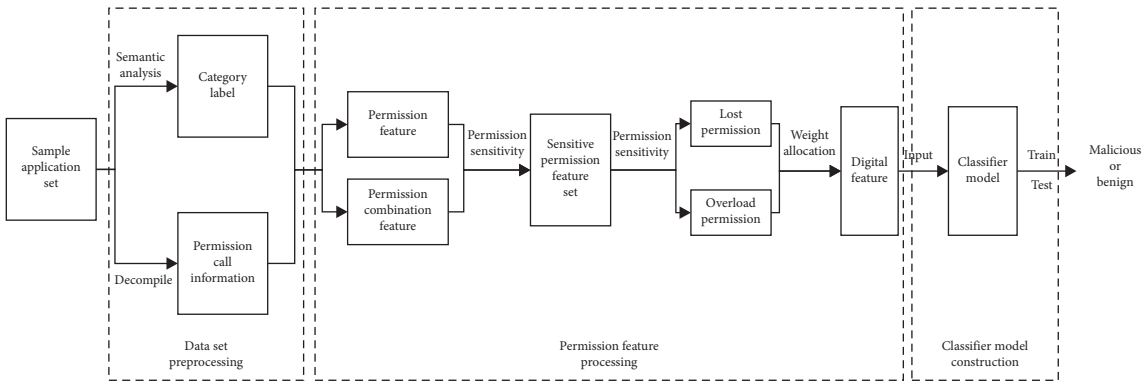


Figure 1: Malicious application detection process.

The average permission matrix of malicious samples is obtained by the same operation:

$$\overline{\mathrm{PM}_{t_iM}} = \frac{\sum_{j=1}^{\mathrm{size}\left(t_iM\right)} \mathrm{PM}_{ti}M_j}{\mathrm{size}\left(t_iM\right)}, \tag{4}$$

size (x) is a function used to calculate the number of elements in the set x. The number in each position of the average permission matrix represents the call frequency of permission features or combination features.

3.1.3. Permission Sensitivity. The average permission matrix of benign samples based on t_i category label is subtracted from that of malicious samples to get the permission sensitivity matrix under the category label. The value of each position in the matrix is the permission sensitivity (PS). Permission sensitivity is a physical quantity to measure the degree of distinguishing between malicious and benign applications. Under the category label t_i, the permission sensitivity of the permission feature m, and the permission sensitivity of the combination feature of permission m and permission n, are respectively expressed as follows:

$$\mathrm{PS}_{t_imm} = \overline{\mathrm{PM}_{t_iB_{mm}}} - \overline{\mathrm{PM}_{t_iM_{mm}}}, \quad 0 \le m \le k, \tag{5}$$

$$\mathrm{PS}_{t_imn} = \overline{\mathrm{PM}_{t_iB_{mn}}} - \overline{\mathrm{PM}_{t_iM_{mn}}}, \quad 0 \le m \le k, 0 \le n \le k, m \le n. \tag{6}$$

The greater the absolute value of the feature's permission sensitivity, the greater the discrimination between benign and malicious applications. For each type of application, permission features and combination features can be filtered to eliminate the redundant features to obtain the sensitive permission feature set based on the permission sensitivity. At the same time, to accurately quantify the application of the sensitive permission feature set, the sensitive permission feature set is further subdivided based on the permission sensitivity, and different types of features are given different weights, which can effectively represent the discriminate strength of features against malicious samples.

3.2. Automatic Classification Based on Semantic Analysis. The same permission request has different legality judgments in various applications, so it is necessary to classify applications to detect malicious applications.

Semantic analysis is used to extract the category label based on application description text. Before submitting an application package to the app store for release, the developer must fill in the application function description text, publisher, and other relevant information. The filling of this information is directly related to the publisher's digital signature. It will be reviewed by the staff of the app store, which has high credibility and authenticity. The specific steps are shown in Figure 2.

The Android application description text obtained by web crawler technology usually contains null characters, special meaning symbols, and so on, which disturb the semantic analysis. It is necessary to use Unicode encoding and other methods to preprocess the application description text to reduce noise. Simultaneously, to classify applications based on description text, it is necessary to predefine the set of category labels.

The preprocessed text set needs to be extracted with feature words. The Term Frequency-Inverse Document Frequency algorithm (TF-IDF) does not involve iterative calculation and word comparison, which saves hardware memory space and extraction time. The effect of feature word extraction is very prominent. Therefore, the TF-IDF algorithm is used to extract feature words from the description text of each type of application [12]. Algorithm 1 gives its pseudocode.

In order to process the relationship between a single feature word and a set of feature words into useful features that can be recognized and read by a machine-learning algorithm, it is necessary to quantize the feature words so that the text information can be transformed into a meaningful number vector or array [13]. Bag of words (BOW) is the most common method of word vectorization. Its main idea is to take a high-order vector whose dimension is equal to the size of the feature word set, and each position of the vector represents the number of times the corresponding word appears in the word set [14]. The word bag model considers the influence of the occurrence times of

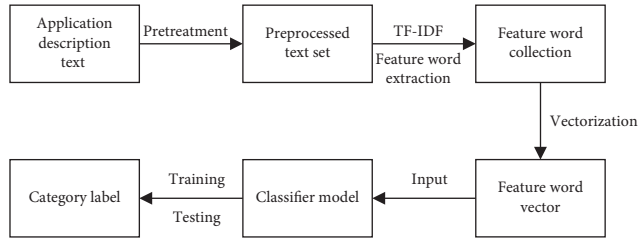


Figure 2: Process of application category determination based on semantic analysis.

Input: T is the label collection, D is the application description text collection
Output: Feature word collection Words
1: initialize an Array of Words
2: **for** each $t_i \in T$ **do**
3: initialize a map Map_{t_i}
4: initialize an array doc_{ij}
5: **for** each $\text{app}_{ij} \in t_i$ **do**
6: $\text{doc}_{ij} = \text{pretreatment}\ (D_{ij})$
7: $\text{words}_{ij} \longleftarrow \text{TF} - \text{IDF}\ (\text{doc}_{ij})$
8: **for** each $\text{word}_k \in \text{words}$ **do**
9: **if** $\text{word}_k \in \text{Map}_{t_i}$ **then**
10: $\text{value}_{\text{word}_k} \longleftarrow \text{value}_{\text{word}_k} + 1$
11: **else** Map_{t_i}.push (key: word_k, value: 1)
12: **end if**
13: **end for**
14: **end for**
15: Words.push (Map_{t_i})
16: **end for**
17: **return** Words

Algorithm 1: Feature word set extraction algorithm of description text based on TF-IDF.

feature words on feature vectors. However, due to the uncertainty of the number and frequency of feature words in a single text, the difference between feature vectors in various texts is too high. The complexity of the subsequent calculation is significantly increased. Therefore, the bag of words model is improved: use the normalized exponential function to process the digital vector based on the bag-of-words model so that the sum of all the digits of the vector is equal to 1. The specific process is as follows:

The extracted feature word set is Words, and the set size is n. Comparing the feature words extracted by a specific application description text d with the feature word set, the vector can be obtained using the bag of words model:

$$\overrightarrow{V_d} = (v_1, v_2, \ldots, v_n)^T. \tag{7}$$

Use the exponential function to normalize the vector:

$$V_{\text{si}} = \frac{e^{vi}}{\sum_{j=1}^{n} e^{vj}}. \tag{8}$$

The normalized vector is

$$\overrightarrow{V_{\text{sd}}} = (v_{s1}, v_{s2}, \ldots, v_{sn})^T. \tag{9}$$

In this way, the feature words can be converted into digital vectors, which can then be input into the classifier model for training and predicting category labels.

3.3. Extraction of Sensitive Permission Feature Set. In order to reduce the detection time and save memory space, the permission features and combination features are filtered based on the permission sensitivity to obtain the sensitive permission feature set based on the category label.

The Android operating system initially designed more than 400 permissions for developers to use. Some permissions can only be called by the system, and ordinary developers have no right to use them. This type of permission is the first to eliminate in the study. Simultaneously, in the latest version of the Android log, it was mentioned that some permissions will rarely be used by the regular application and will be discarded in future versions, such as the SET_PREFERRED_APPLICATIONS permission, so these permissions are filtered. Use $P_{\text{er}} = \{P_{\text{er1}}, P_{\text{er2}}, \ldots, P_{\text{erk}}\}$ to indicate the remaining set of permissions after filtering.

The benign application samples B_{t_i} and malicious application samples M_{t_i} under the label t_i are obtained from the public data set. Related operations such as decompiling the APK file are carried out to get the application's permission call information. For a single application sample app_{ij} under the t_i category label, the binary permission vector $\overrightarrow{\text{PV}_{t_{ij}}}$ is

obtained by comparing the permission call information with the permission set.

The transposition vector of the feature vector is denoted as $\mathrm{PV}_{t_{ij}}T$, according to formula (2), the feature matrix $\underrightarrow{\mathrm{PM}}_{t_i j}$ of $k \times k$ dimension is obtained by multiplying with $\overrightarrow{\mathrm{PV}}_{t_{ij}}$.

According to formulas (3) and (4), the average permission matrix $\underrightarrow{\mathrm{PM}}_{t_i B}$ of benign samples and the average permission matrix $\underrightarrow{\mathrm{PM}}_{t_i M}$ of malicious samples can be calculated. The diagonal number in the average permission matrix represents the permission coverage of the corresponding permission feature. The nondiagonal number represents the permission coverage of the corresponding permission combination feature. Permission coverage is a measure of the frequency of calling a specific permission feature or permission combination feature under a single category.

When the average permission matrix of benign application samples is subtracted from that of malicious application samples under the t_i category label, the value is the permission sensitivity of corresponding features. According to formula (5), PStimm, the sensitivity of permission feature m to malicious samples and benign samples can be calculated. According to formula (6), $\mathrm{PS}_{t_i mn}$, the sensitivity of permission combination features of permission m and permission n under the t_i category label can be calculated.

The permission sensitivity indicates the ability of the feature to distinguish between benign and malicious applications. The higher the absolute value of permission sensitivity, the more pronounced the discrimination between benign and malicious applications. Therefore, for all the permission features and permission combination features under the t_i category label, the permission sensitivity $x_1, x_2, \ldots, x_n$ is arranged from large to small according to the absolute value. Set threshold μ:

$$\mu = \min_x \frac{\sum_{j=1}^{n} \left(x - x_j\right)^2}{(n-1)^2}. \tag{10}$$

The features whose absolute value of permission sensitivity is greater than or equal to the threshold are put into the sensitive permission feature set, which means the permission feature can distinguish benign and malicious applications more effectively to increase the detection accuracy. If the absolute value of the permission sensitivity is less than the threshold value, the feature which wastes time and space resources will be eliminated. When faced with conflicts between permission features and permission combination features, the permission combination features are considered to be more effective in distinguishing malicious and benign, and the permission combination features are added to the sensitive permission feature set, while the permission features are discarded, and the sensitive permission feature set of the t_i category label is obtained. After repeated operations, the sensitive permission feature set corresponding to all category labels can be obtained.

3.4. Quantification of Sensitive Permission. The permission request information of the application to be tested is compared with the sensitive permission feature set based on the category label. The result is represented by digital features, which better reflects the malicious degree of the application. This process is the quantification of sensitive permission features.

In traditional malware detection methods, binary methods are usually used to process features. Use 0 to indicate that the permission is not invoked, and 1 indicates that the permission is invoked. Thus the high-dimensional vectors of 0 and 1 are formed and input into the classifier. But the binary method has apparent shortcomings, for the permission group set per, assuming that the sensitivity of permission i is greater than that of j, it means that the distinguishing power of permission feature i for malicious and benign samples is higher than that of permission feature j. If expressed by the binary method, their corresponding positions will be assigned as 1 or 0, which cannot reflect the degree of distinguishing malicious and benign applications. The neglect of the relationship will affect the detection results, so the weight allocation method is proposed to quantify the features.

First, two types of permissions are defined: lost permission and overload permission.

3.4.1. Lost Permission. When calculating the permission sensitivity, the values are positive and negative. When the permission sensitivity of features is positive, under the category label, the permission coverage of benign application samples is greater than that of malicious application samples. Such features are defined as lost permissions (LP). Under the category label, the probability that the permission feature j appears in the benign application sample is higher than that in the malicious application sample. The benign application usually calls the permission feature j, while the malicious application rarely calls the permission feature j, which is the lost permission.

3.4.2. Overload Permission. The permission features with negative permission sensitivity under the same category label. That means the probability of permission feature j appearing in the malicious application sample is higher than that in the benign application sample. The malicious application usually calls permission feature j, while benign application rarely calls permission feature j, which is overload permission.

Sensitive permissions are divided into two categories because they have different influences on malware detection. The lost permission means that the application under test does not call one or more key permission features than benign applications. For example, a photographing application does not apply for calling camera permission. Under normal circumstances, it can be considered that the application under test may not have the normal functions of other applications under the category label. Still, there will be no other malicious behavior. Overload permission means that the application under test calls one or more key permission features, such as a shopping application calling contact information, compared with the benign application. This

extra behavior should be considered as malicious behavior beyond the acceptable range. Therefore, the discrimination between malicious and benign of the overload permission should be higher than that of the lost permission. When α and β are used to measure the discrimination of overload permission and the lost permission for malicious samples, $\alpha \geq \beta$.

The main disadvantage of the binary method is that it cannot reflect the difference in permission sensitivity. Therefore, all permission features under the label are normalized according to the sensitivity. Each permission feature is assigned a weight to measure the ability to distinguish benign and malicious application samples. Assuming that the sensitive permission feature set based on the t_i category label is $(P_1, \mathrm{PS}_{t_i 1})$, $(P_2, \mathrm{PS}_{t_i 2}), \ldots$, $(P_n, \mathrm{PS}_{t_i n})$, where P is the specific permission, and PS represents the sensitivity of the permission. For permission P_k, the weight can be calculated as follows:

$$w_k = \frac{\left|\mathrm{PS}_{t_i k}\right|}{\sum_{m=1}^{n}\left|\mathrm{PS}_{t_i m}\right|}. \tag{11}$$

For all features in the sensitive permission feature set, the weight vector W is obtained:

$$\overrightarrow{W} = (w_1, w_2, \ldots, w_n)^T. \tag{12}$$

Combining overload permissions and lost permissions, the above formula is further processed:

$$\overrightarrow{\mathrm{PW}} = (\mathrm{pw}_1, \mathrm{pw}_2, \ldots, \mathrm{pw}_n)^T,$$
$$\mathrm{pw}_i = \begin{cases} w_i \times \alpha, & p_i \in \mathrm{OP}, \\ w_i \times \beta, & p_i \in \mathrm{LP}. \end{cases} \tag{13}$$

Assuming that the application permission call vector obtained by the binary method is $\overrightarrow{Bin_i}$, then the digital feature obtained by the weight distribution method is

$$W_i = \overrightarrow{Bin_i} \cdot \overrightarrow{\mathrm{PW}}. \tag{14}$$

The pseudocode of the specific weight allocation algorithm is shown in Algorithm 2.

3.5. Construction of Classifier Model for Malware Detection. A classifier is constructed to process the digitized sensitive features to complete the malware detection based on the machine-learning algorithm. Among the most commonly used traditional machine-learning algorithms, the random forest algorithm has excellent performance compared to other traditional machine-learning algorithms [15]. It takes less time to process large-scale samples. Therefore, the random forest algorithm is used to construct a classifier model for Android malware detection. The detailed process is shown in Algorithm 3.

4. Experimental Results and Analysis

In this experiment, 3400 Android malicious application samples from the VirusShare [16] and Arp et al. [17] data sets, 4500 normal applications as the benign Android samples were selected from the Huawei application market through crawler technology. 80% of the samples are used as the training set, and 20% of the samples are used to test the classifier model's performance. Before classifying the applications, the application category label set is predefined: navigation, online shopping, weather, finance, education, catering, camera, music, social, communication, news, video, reading, tourism, sports, office, tools, and game.

4.1. Evaluation Metrics. Suppose TP and TN are the numbers of benign applications and malicious applications correctly identified, FP and FN are the numbers of misjudged malicious applications and benign applications, respectively. Then the commonly used evaluation metrics of two classification problems are accuracy A, precision P_{re}, recall R_{ec} and F_1:

$$A = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}},$$
$$P_{\mathrm{re}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}},$$
$$R_{\mathrm{ec}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{15}$$
$$F_1 = \frac{2 \times \mathrm{TP}}{2 \times \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}.$$

For the multiclassification problem, the concept of macro average is introduced, and the evaluation metrics of each category label are calculated by arithmetic average: macro precision P_{macro}, macro recall R_{macro}, and macro $F_{1\mathrm{macro}}$ to measure the performance of the detection method.

$$P_{\mathrm{macro}} = \frac{1}{n}\sum_{i=1}^{n} P_i,$$
$$R_{\mathrm{macro}} = \frac{1}{n}\sum_{i=1}^{n} R_i, \tag{16}$$
$$F_{1\mathrm{macro}} = \frac{2 \times P_{\mathrm{macro}} \times R_{\mathrm{macro}}}{P_{\mathrm{macro}} + R_{\mathrm{macro}}}.$$

The classifier model includes two processes: training and testing. Symbol t_r is used to represent the time taken by the classifier to complete the sample training, the symbol t_p to represent the time taken for the classifier to complete the sample test, the symbol t_s to represent the total time for training and prediction, and the symbol $\bar{t}$ to represent the time consumed for processing a single sample.

4.2. Analysis of Results. In the experiment of category label determination, k-nearest neighbor (KNN) algorithm [18], Naive Bayes Model (NBM) [19], and TextCNN algorithm [20] are used to construct classifier models for comparison. In the malware detection experiment, the value of α/β in the model is tuned. The optimized detection model is compared with the traditional detection model to check accuracy and efficiency.

Input: Test is the application to be tested
Output: The weight of permission
for each app ∈ Test **do**
  initialize a value result = 0
  $\text{Bin}_{\text{app}} \longleftarrow$ Binary (app)
  for each $i \in \text{PS}_{\text{Groups}}$ **do**
    $W_i \longleftarrow$ Normalized (W_i)
    if $i \in$ LP **then**
      result $\longleftarrow$ result + $\text{Bin}_{\text{app}_i} * W_i * \alpha$
    else
      result $\longleftarrow$ result + $\text{Bin}_{\text{app}_i} * W_i * \beta$
    end if
  end for
  return result
end for

Algorithm 2: The indication of sensitive permission call information based on the weight allocation method.

Input: data is the sample data set, s is the data set extraction ratio, DT is the decision tree collection, x is the attribute classification method
Output: result Malware detection result
initialize a value num_b
initialize a value num_m
for each $dt \in$ DT **do**
  initialize a Boolean temp
  $\text{data}_{dt} \longleftarrow \text{data} * s$
  temp$\longleftarrow$DT(data_{dt}, x)
  if temp = true **then**
    $\text{num}_b \longleftarrow \text{num}_b + 1$
  end if
  if temp = false **then**
    $\text{num}_m \longleftarrow \text{num}_m + 1$
  end if
end for
if $\text{num}_b > \text{num}_m$ **then**
  result$\longleftarrow$Benign
end if
if $\text{num}_b < \text{num}_m$ **then**
  result$\longleftarrow$Malicious
end if
return result

Algorithm 3: Construct a classifier based on the random forest algorithm to perform malware detection.

4.2.1. k-Nearest Neighbor Classifier. The accuracy of the classifier model constructed by the k-nearest neighbor algorithm mainly depends on two aspects: the value of k and the choice of distance calculation methods. In this experiment, these two parameters were tuned separately, and the results are shown in Figure 3.

Figure 3(a) shows the evaluation metrics A, P_{macro}, R_{macro}, and $F_{1\text{macro}}$ of KNN classifier model using Euclidean distance under different k values. It can be seen that when $k < 15$, the performance of classifier accuracy is generally low and fluctuates significantly at that time. When $k = 15$, all the classifier's evaluation indexes reached the best, which were 0.8842, 0.9016, 0.8749, and 0.8853, respectively. When $k > 15$, the indicators of the model declined slowly. Therefore, when k is 15, the k-nearest neighbor classifier model has the best performance.

Figure 3(b) shows the k-nearest neighbor classifier model's performance results using Euclidean distance and Manhattan distance, respectively. From the graph, we can find that the A, P_{macro}, R_{macro}, and $F_{1\text{macro}}$ of the classifier using Manhattan distance formula are generally low, which is far less than the classifier's performance using Euclidean distance under the same conditions. According to the above experimental results, when the k-nearest neighbor classifier model has the best performance, the value of k should be 15, and the distance calculation method should be Euclidean distance.

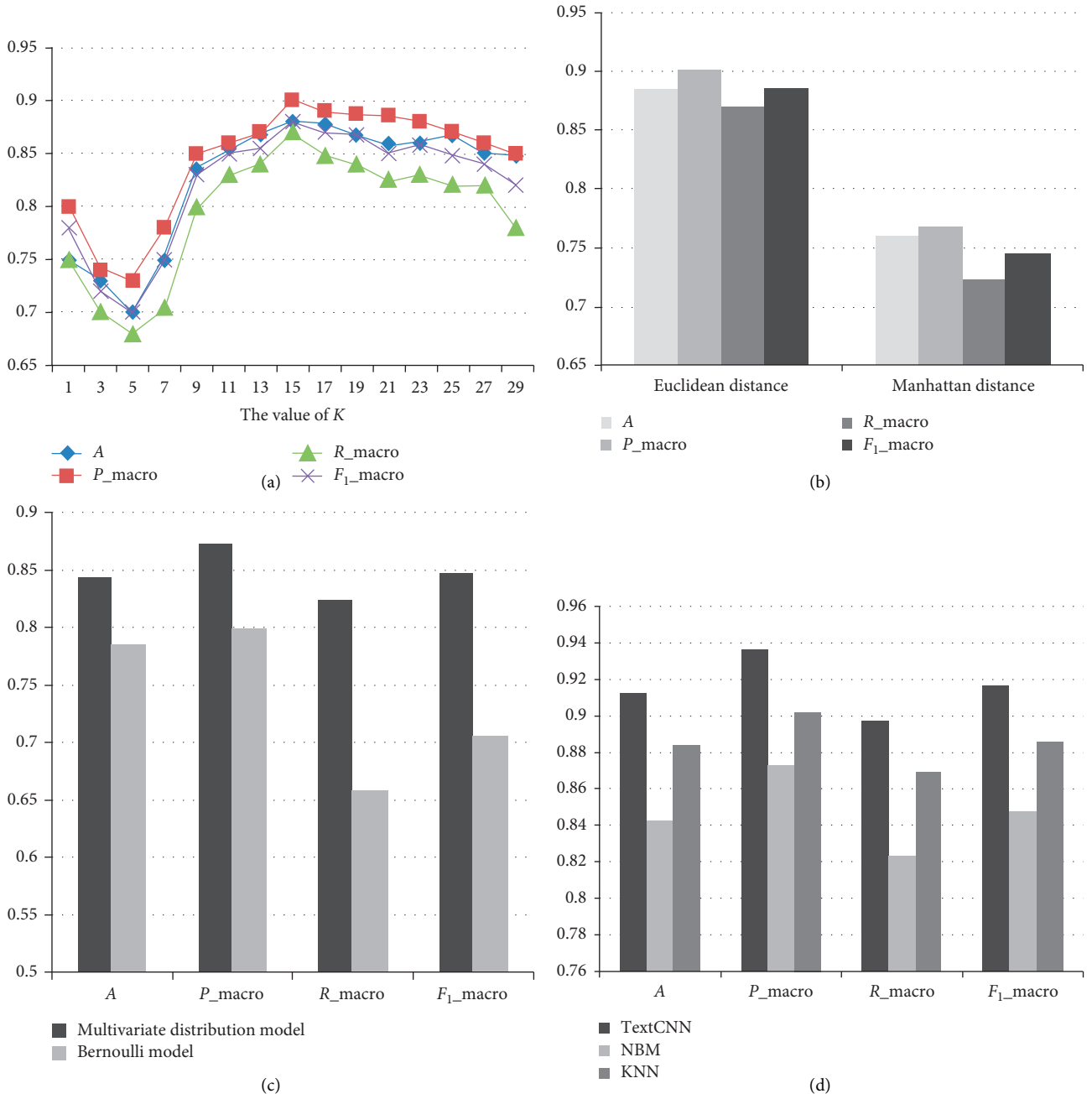


Figure 3: Experimental results of different classifiers. (a) KNN classifier with different k values, (b) KNN classifier with different distance formulas, (c) NBM classifier under different models, and (d) performance of different classifier models.

4.2.2. Naive Bayes Model Classifier. The classifier model's parameters based on the Naive Bayes Model are optimized to explore the distribution model (Bernoulli Model, Multivariate Distribution Model), which makes the accuracy performance of the classifier model the best. The results are shown in Figure 3(c). It can be seen from Figure 3(c) that under the same data set, the evaluation metrics obtained using the multivariate distribution model are better than those using the Bernoulli model, which is of great help to improve the accuracy performance of the classifier.

4.2.3. Performance Comparison of Three Classifier Models. TextCNN algorithm is a representative algorithm model to solve the problems of text classification and semantic analysis. Compared to the k-nearest neighbor algorithm classifier model, the Naive Bayes algorithm classifier model, and the TextCNN algorithm classifier model with the performance, the results are shown in Figure 3(d). The graph shows that the A, P_{macro}, R_{macro}, and $F_{1\text{macro}}$ of the classifier constructed based on the TextCNN algorithm are 0.9165, 0.9368, 0.8971, and 0.9127, respectively, which are better than the other two models. It can be seen that in the data sets

Table 1: Efficiency performance of three classifier models.

	t_r (s)	t_p (s)	t_s (s)	$\bar{t}$ (s)
KNN	2202.81	339.63	2542.44	0.2472
NBM	1226.97	308.74	1535.71	0.1677
TextCNN	2474.62	1213.95	3688.57	0.5090

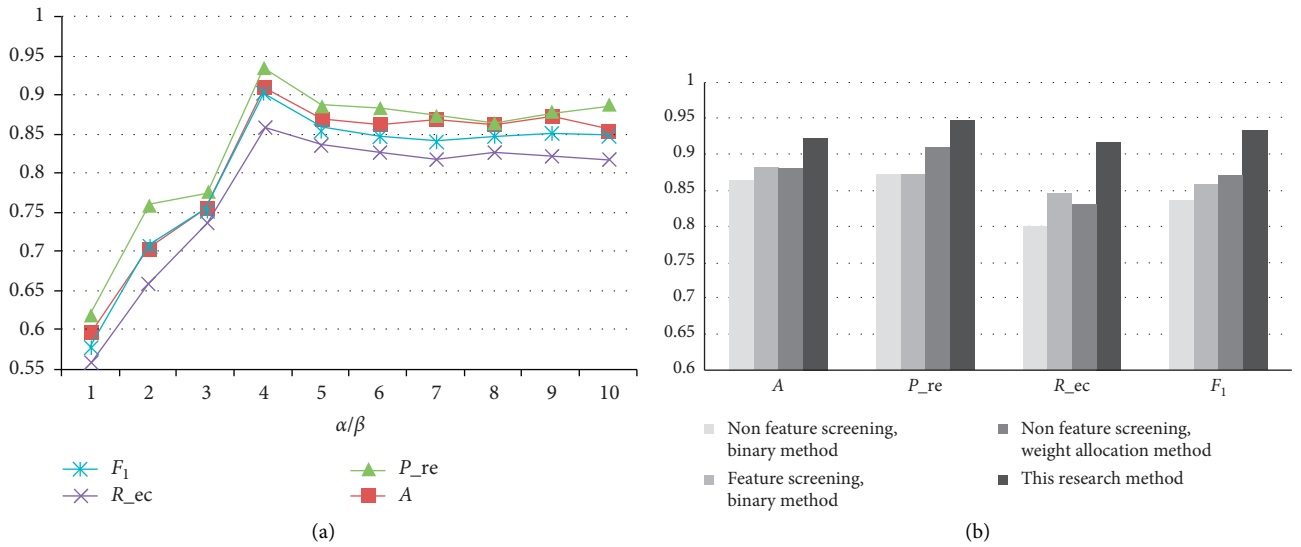


Figure 4: Experimental results of malware detection. (a) α/β optimization experiment results and (b) accuracy performance results of four methods.

and application scenarios of this study, the deep learning algorithm is more suitable than the traditional machine-learning algorithm in constructing a classifier model to complete the automatic application category determination based on semantic analysis.

Table 1 shows the performance of the three models on classification speed. It can be seen from Table 1 that in the total training and prediction time, the TextCNN classifier model with the best accuracy performance takes the most time, reaching 3688.57 seconds. In terms of the average training and prediction time of a single sample, the Naive Bayesian Model performs best, and it only takes 0.1677 seconds to process a sample. Considering that the ratio of training data set to test data set in this experiment is 4 : 1, from the approximate results, the ratio of training time to test time of classifier constructed based on Naive Bayesian algorithm is equal to the ratio of training data set to training data set, that is, the time complexity can be considered as $O(n)$. Although the total time consumed by the TextCNN model is relatively large, the training and prediction time is relatively large, and the time complexity is similar to $O(n)$. This means that when the data set is vast, the disadvantage of using the TextCNN algorithm to construct a classifier will be significantly reduced, which can even become a more efficient algorithm.

In summary, the accuracy of the three algorithms for application classification is above 85%. In terms of accuracy, the classification accuracy based on the TextCNN algorithm reached 91.65%, and the macro precision reached 93.68%, which is better than the other two classifier models. In terms of efficiency, the classifier model based on the Naive Bayes model takes only 0.1677 seconds to process a single sample, which is suitable for application stores with high-efficiency classification.

4.2.4. Determination of Parameters in the Malware Detection Model. In quantifying sensitive permission features based on weight allocation, the ratio measures the ability of permission features to distinguish malicious and benign samples. Figure 4(a) shows the results of the accuracy rate A, precision rate P_{re}, recall rate R_{ec}, and F_1 value of the malware detection method as the parameters $\alpha\beta$ change. It can be seen from the figure that when the parameter is small, especially when $\alpha\beta = 1$, the classifier's evaluation metrics are not ideal, and the detection accuracy of malicious applications is only 59.76%. As the parameter $\alpha\beta$ gradually increase, each evaluation metric curve rises with a larger slope, and the detection performance is significantly improved. When $\alpha\beta = 4$, each evaluation metric of the classifier reaches the maximum value. Therefore, the optimal value of the $\alpha\beta$ should be 4.

4.2.5. Performance Comparison of Four Different Malicious Detection Models. The Android malware detection method proposed in this paper is compared with the detection model without feature screening and using the binary method, the detection model with feature screening and using the binary method, and the detection model without feature screening and using the weight allocation method to compare the

Table 2: Efficiency performance of four detection models.

	t_r (s)	t_p (s)	t_s (s)	$\overline{t}$ (s)
This research method	423.27	94.16	517.43	0.1256
Nonfeature screening, binary method	903.75	286.37	1190.12	0.3217
Feature screening, binary method	392.56	106.12	498.68	0.1282
Nonfeature screening, weight allocation	883.24	230.69	1113.93	0.3012

accuracy performance and efficiency performance of the four methods. Figure 4(b) shows the results of these four methods on accuracy rate A, precision rate, recall rate, and F_1 value. It can be seen from the figure that the accuracy rate of the malicious detection method proposed in this research reached 92.17%, and the precision rate reached 94.68%. It is obviously better than the other three methods in each index. Replacing the binary method with the weight allocation method, replacing the malicious application detection method by obtaining the sensitive permission feature set can increase the fine-grained of malware detection and improve malware detection accuracy.

Table 2 shows the efficiency performance of the above four methods in training time, prediction time, total time, and average sample time. In the case of feature screening, the weight allocation method and the binary method are almost the same in each time index. When the sensitive permission feature set is not extracted, the redundant features are too much, and the measured time is significantly increased. This shows that the method proposed in this paper successfully reduces the dimension of digital features and improves malicious detection efficiency. Compared with the traditional method of direct detection of unfiltered permission features, the proposed method reduces the time by 60.94%.

To sum up, the detection method for malicious Android applications proposed in this paper based on permission sensitivity has an accuracy rate of 92.17%, 60.94% less than the traditional detection method.

5. Conclusion

In this paper, an Android malware detection method based on permission sensitivity is proposed. Based on the application category label and permission request information, the concepts of permission vector, permission matrix, and permission sensitivity are established, and the sensitive permission feature set corresponding to each category label is obtained by mathematical formula reasoning. Through the weight allocation method, each permission feature is given a weight value that measures the strength of distinguishing malicious and benign samples to form a digital feature. The classifier constructed by the random forest algorithm is used for malicious detection, and multiple sets of experiments are set for verification and comparison.

The experimental results show that the accuracy rate of the Android malware detection method can reach 92.17%, and the precision rate can reach 94.68%. Compared with the traditional detection method based on permission features, the time consumption of the malware detection method is reduced by 60.94%.

Data Availability

The data in this paper are divided into benign samples and malicious samples. The malicious samples were taken from the VirusShare and Drebin data sets. The benign samples are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partly supported by the National Natural Science Foundation of China (grant no. 61601113) and the National Key R&D Program of China (grant nos. 2018YFB2202200 and 2018YFB2100403).

References

[1] T. Wu and Y. Yang, "Detecting android inter-app data leakage via compositional concolic walking," *Intelligent Automation & Soft Computing*, vol. 25, no. 4, pp. 755–766, 2019.

[2] R. Song, Y. Song, Q. Dong, A. Hu, and S. Gao, "Weblogger: stealing your personal pins via mobile web application," in *Proceedings of the 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, Nanjing, China, October 2017.

[3] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for Android," in *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices (SPSM'11)*, pp. 15–26, Association for Computing Machinery, Chicago, IL, USA, August 2011.

[4] C. Yuan, S. Wei, Y. Wang, Y. You, and S. G. ZiLiang, "Android applications categorization using bayesian classification," in *Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 173–176, Chengdu, China, October 2016.

[5] A. Kutlay and K. Karaduzovic-Hadziabdic, "Static based classification of malicious software using machine learning methods," in *Proceedings of the Advanced Technologies, Systems, and Applications IV -Proceedings of the International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies (IAT 2019)*, Sarajevo, Bosnia and Herzegovina, June 2019.

[6] F. Ahmad, A. N. Badrul, K. Ahmad et al., "Discovering optimal features using static analysis and a genetic search based method for Android malware detection," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 6, pp. 712–736, 2018.

[7] W.-C. Wu and S.-H. Hung, "DroidDolphin: a dynamic Android malware detection framework using big data and machine learning," in *Proceedings of the 2014 Conference on Research in Adaptive and Convergent Systems (RACS'14)*, pp. 247–252, Association for Computing Machinery, Towson, MD, USA, October 2014.

[8] J. Jeon, J. H. Park, and Y.-S. Jeong, "Dynamic analysis for IoT malware detection with convolution neural network model," *IEEE Access*, vol. 8, Article ID 96899, 2020.

[9] M. Lindorfer, M. Neugschwandtner, and C. Platzer, "MARVIN: efficient and comprehensive mobile app classification through static and dynamic analysis," in *Proceedings of the IEEE 39th Annual Computer Software and Applications Conference*, pp. 422–433, Taichung, Taiwan, July 2015.

[10] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A multimodal deep learning method for android malware detection using various features," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 773–788, March 2019.

[11] Y. Li, G. Xu, H. Xian, L. Rao, and J. Shi, "Novel android malware detection method based on multi-dimensional hybrid features extraction and analysis," *Intelligent Automation and Soft Computing*, vol. 25, no. 3, pp. 637–647, 2019.

[12] Z. Zhou, J. Qin, X. Xiang, Y. Tan, Q. Liu, and N. Xiong, "News text topic clustering optimized method based on tf-idf algorithm on spark," *Computers, Materials & Continua*, vol. 62, no. 1, pp. 217–231, 2020.

[13] L. Yan, Y. Zheng, and J. Cao, "Few-shot learning for short text classification," *Multimedia Tools and Applications*, vol. 77, no. 22, Article ID 29799, 2018.

[14] N. Chayangkoon and A. Srivihok, "Feature reduction of short text classification by using bag of words and word embedding," *International Journal of Control and Automation*, vol. 12, no. 2, pp. 1–16, 2019.

[15] D.-W. Kim, G.-Y. Shin, and M.-M. Han, "Analysis of feature importance and interpretation for malware classification," *Computers, Materials & Continua*, vol. 65, no. 3, pp. 1891–1904, 2020.

[16] "Virus share [EB/OL]," https://virusshare.com/.

[17] D. Arp, M. Spreitzenbarth, M. Hubner et al., "Drebin: effective and explainable detection of android malware in your pocket," *Ndss*, vol. 14, pp. 23–26, 2014.

[18] M. Anshori, I. F. Mar, and F. A. Bachtiar, "Comparison of machine learning methods for android malicious software classification based on system call," in *Proceedings of the International Conference on Sustainable Information Engineering and Technology (SIET)*, Lombok, Indonesia, September 2019.

[19] P. R. K. Varma, K. P. Raj, and K. V. S. Raju, "Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms," in *Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 294–299, Palladam, India, February 2017.

[20] X. Qin, S. Peng, X. Yang, and Y.-D. Yao, "Deep learning based channel code recognition using TextCNN," in *Proceedings of the IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–5, Newark, NJ, USA, November 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9916461, 9 pages
https://doi.org/10.1155/2021/9916461

Research Article

Meteorological Satellite Operation Prediction Using a BiLSTM Deep Learning Model

Yi Peng,[1] **Qi Han**,[1] **Fei Su**,[2] **Xingwei He**,[1] **and Xiaohu Feng**[1]

[1]*National Satellite Meteorological Center, Beijing 100081, China*
[2]*China Unicom Smart Connection Technology Co., Ltd, Beijing 100037, China*

Correspondence should be addressed to Xiaohu Feng; fengxh@cma.gov.cn

Received 13 March 2021; Accepted 31 May 2021; Published 21 June 2021

Academic Editor: Liguo Zhang

The current satellite management system mainly relies on manual work. If small faults cannot be found in time, it may cause systematic fault problems and then affect the accuracy of satellite data and the service quality of meteorological satellite. If the operation trend of satellite will be predicted, the fault can be avoided. However, the satellite system is complex, and the telemetry signal is unstable, nonlinear, and time-related. It is difficult to predict through a certain model. Based on these, this paper proposes a bidirectional long short-term memory (BiLSTM) deep leaning model to predict the operation trend of meteorological satellite. In the method, the layer number of the model is designed to be two, and the prediction results, which are forecasted by LSTM network as the future trend data and historical data, are both taken as the input of BiLSTM model. The dataset for the research is generated and transmitted from Advanced Geostationary Radiation Imager (AGRI), which is the load of FY4A meteorological satellite. In order to demonstrate the superiority of the BiLSTM prediction model, it is compared with LSTM based on the same dataset in the experiment. The result shows that the BiLSTM method reports a state-of-the-art performance on satellite telemetry data.

1. Introduction

Meteorological satellite in a complex space environment that contains all kinds of loads is a complicated system, whose components have coupling relationship. Abnormality and faults easily occur. Small faults may cause systematic fault problems, then affect the accuracy of satellite data and service accuracy, and even cause significant economic losses and serious disasters to people's livelihood and the country. If the operation trend of satellite will be predicted, the fault can be avoided. It is of great significance for the intelligent health management of meteorological satellite and has reference value for the health management of other satellites.

At present, the trend prediction methods mainly include statistical prediction, mathematical prediction, intelligent prediction, and information fusion prediction. Among them, Auto Regression Moving Average (ARMA) prediction model, Support Vector Regression (SVR) prediction model, Back Propagation Network (BP) prediction model, and Long Short-Term Memory (LSTM) prediction model are widely studied and applied. ARMA prediction model is a linear model with finite parameters. For short-term prediction, the model has high fitting accuracy [1, 2]. But it is not suitable for nonlinear and nonstationary sequences. SVR prediction model based on structural risk minimization criterion has better prediction generalization ability for small sample training set, higher prediction accuracy, and better robustness than ARMA model [3]. But it can only be used as short-term prediction algorithm. In recent years, SVR is widely used to solve practical problems in various fields by combining with other algorithms [4–6]. BP network has become one of the most widely used neural network models because of its strong nonlinear mapping ability and self-learning ability [7–9]. However, it has some problems, such as slow convergence speed, lack of scientific theoretical basis for the determination of hidden layer nodes, and easily falling into local minimum points [10]. Deep learning is a branch of neural networks. The recurrent neural network

(RNN) with depth and time series has been widely used as a prediction model for sequence data [11–13]. Although RNN can deal with time series problem, it has serious gradient dispersion problem. In order to weaken the adverse effects of gradient disappearance and long-distance dependence on neural network, the long-term memory ability of RNN is improved by replacing RNN chain unit with LSTM chain unit [14]. LSTM that uses additional memory cell to store states shows better ability for time series prediction than RNN. In recent years, LSTM has achieved a significant success in many fields [15–17].

Although LSTM model has good prediction ability of nonlinear time series [18], it takes historical time series data as the data input of the model, neglects the availability of future time series data, and lacks deep data mining. Moreover, the telemetry signal generated and transmitted from the satellite that reflects the operation of the satellite is unstable, nonlinear, and time-related. It is difficult to predict the satellite operation through a certain model. Bidirectional Long Short-Term Memory (BiLSTM) can process the sequence data in both forward and reverse directions and provide the past and future sequence information for each time in the sequence [19]. Therefore, BiLSTM mode is proposed to predict the operation trend of satellite in this paper. Increasing the number of depth layers of neural network can help enrich the feature set of network learning, increase the processing capacity of neural network, and improve the accuracy of the model. In addition, the paper presents using LSTM to predict the future trend and take the prediction results and historical data as the input of BiLSTM in order to provide the future time series data.

The remainder of this paper is organized as follows. In the next section, a brief description of problem definitions is presented, followed by the introduction of the proposed BiLSTM network architecture. Then, the application to meteorological satellite based on BiLSTMdeep learning model and our experimental results are shown. Finally, conclusions are drawn.

2. Problem Definitions

Satellite management plays an important role for satellite safe. However, the current satellite management mainly relies on manual work, which takes time and effort. And these can lead to the fact that fault cannot be found in time. This paper intends using artificial intelligence algorithm to solve the current problems, by deeply mining and analyzing telemetry data, study the automatic prediction method for the operation trend of meteorological satellite, and solve the problems that restrict the stable operation of satellite and abnormalities, which cannot be found in time.

Based on the safety and reliability requirements of satellite, when it is on orbit, sensors are designed in the main functional modules of each key subsystem during satellite development. These sensors provide telemetry parameters data of satellite from the launch to in-orbit operation to the retirement. The satellite telemetry system collects the working conditions and values of various subsystems on the satellite according to a certain sampling period and forms telemetry data after A/D transformation and coding. And then, it transmits them to the ground through modulation and amplification, and these telemetry data are gotten through the inverse process after receiving them on the ground. Satellite telemetry data are divided into two categories: digital quantity and analog quantity. The digital quantity reflects the functional state of the measured unit on the satellite. Analog is the numerical measurement value of the measured unit, which usually reflects the performance state of the measured unit. Satellite telemetry parameters include thermodynamic parameters, power system parameters, and dynamic parameters. It is no doubt that a large amount of data will be generated. These telemetry data reflect the state of satellite payload and the operation situation of satellite platform, which are stored in time series. Therefore, the operation trend for meteorological satellite is related to an amount of spatial-temporal data.

The telemetry data contain a lot of objective laws and knowledge that can be used for trend prediction. The operation trend of satellite platform and load can be predicted based on these heterogeneous, coupled, and large-scale telemetry data (Figure 1 shows some parameters). However, the research faces the following challenges:

(1) Feature representation is difficult: there are thousands of satellite telemetry data variables; telemetry data have problems such as outliers and uneven time intervals in telemetry data; due to the coupling and correlation of satellites, it is difficult for a single parameter to describe the comprehensive performance, and to determine which parameters can accurately describe a certain performance.

(2) It is difficult to predict the trend: the satellite system is complex, the telemetry signal is nonstationary and nonlinear, and the telemetry parameters have three different variation patterns: stationary, abrupt, and periodic.

Facing the difficult of feature representations, this paper selects key features according to the contribution of the operation trends of satellite load and the relationship of the features. Through the analysis of the traditional machine learning algorithm for time series learning prediction, the traditional algorithms are no longer suitable for the operation trend prediction of complex meteorological satellite. Therefore, this paper will build a new prediction model for satellite telemetry data.

The central problem in the model of predicting satellite operation can be represented in the following terms. The problem of satellite operation can be defined using the spatiotemporal variable sequence of $X_{t\text{-}T}$ for prediction. The model can be denoted by

$$Y_{t+T} = f\left(X_t, X_{t-T}\right), \tag{1}$$

where $Y_t + _T$ is the predicted object in next T hours, f represents the final model learnt by the historical data, X_t denotes the datasets at the predicting moments, and $X_{t\text{-}T}$ are the datasets in T hours before the predicting moments.

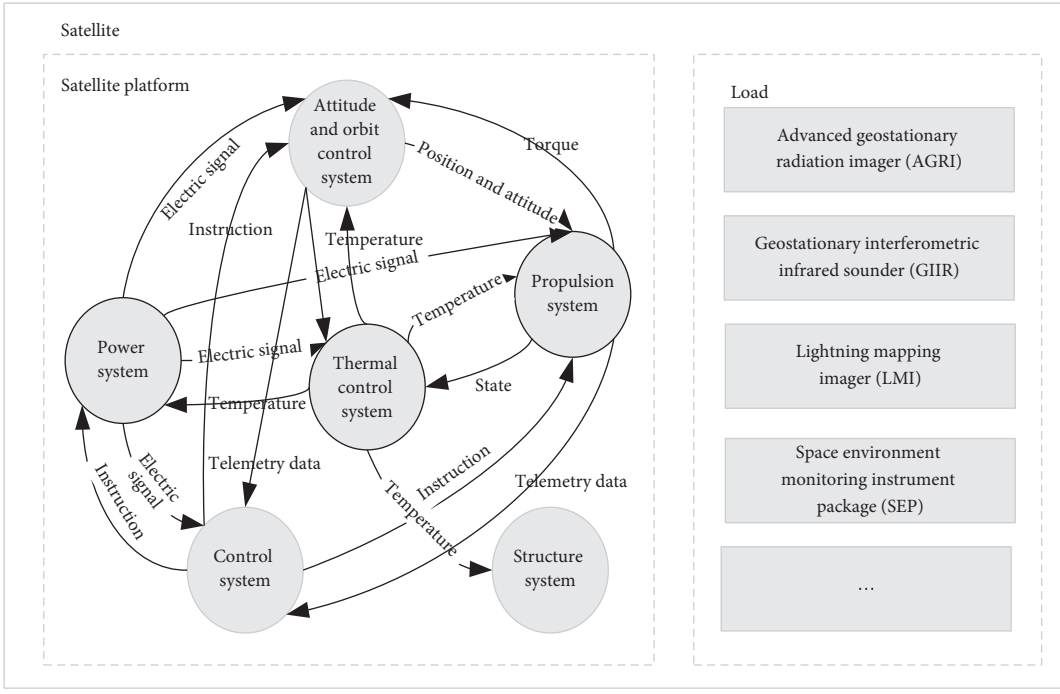


Figure 1: The sketch of complex satellite system.

3. The Proposed Scheme

3.1. The Introduction of LSTM Model. Compared with the traditional RNN, LSTM not only has a hidden state, but also adds a cell state. At the same time, it adds input gate, output gate, and forgetting gate in each layer to control the degree of adding or deleting information. The model can learn the long-term dependence information and avoid the problem of gradient disappearance [20, 21], with smaller error and higher prediction accuracy. At present, the most widely used LSTM network is to use the LSTM unit to replace the neural nodes in the hidden layer of RNN [22]. Its structure is shown in Figure 2, where c_{t-1} denotes the cell state of the previous moment, h_{t-1} is the output of the former LSTM, x_t and h_t represent the input and state output for the current moment respectively, and f_t, i_t and o_t represent the output value of the forgetting gate, the input gate, and the output gate, respectively. The LSTM unit update process is as follows:

$$\tilde{c}_t = \tanh\left(W_c \cdot \left[h_{t-1}, x_t\right] + b_c\right), \tag{2}$$

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right), \tag{3}$$

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right), \tag{4}$$

$$c_t = f_t c_{t-1} + i_t \tilde{c}_t, \tag{5}$$

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_c\right), \tag{6}$$

$$h_t = o_t \tanh\left(c_t\right), \tag{7}$$

where $\tilde{c}_t$ and c_t represent the candidate cell status and the current cell state, respectively. Wc, W_f, W_i and W_o represent the weight of the candidate input gate, the forget gate, the input gate, and the output gate, respectively. b_c, b_f, b_i and b_o represent the bias of the candidate input gate, the forget gate, the input gate, and the output gate, respectively. σ and tanh represent sigmoid activation function and hyperbolic tangent activation function, respectively.

The training process of LSTM model adopts BPTT algorithm, which is similar to the classic back propagation (BP) algorithm. It can be roughly divided into four steps:

(1) Calculating output value of LSTM cells according to forward calculation method (formulas (2)–(7))

(2) Reversely computing error terms for each LSTM cell, which include two reverse propagation directions by time and network level

(3) Calculating the gradient of each weight according to the corresponding error term

(4) Updating weights based on the Optimization Algorithm of Gradient

There are many kinds of gradient-based optimization algorithm, such as Stochastic Gradient Descent (SGD) [23], Adaptive Gradient (AdaGrad) [24], Root Mean Square Prop (RMSProp) [25], and Adaptive Moment Estimation (Adam) [26]. Under the same number of iterations, the convergence speed of Adam algorithm is better than that of other algorithms. Furthermore, the loss function of Adam algorithm has the lowest value, and it costs the least time in all these stochastic optimization methods. Therefore, this paper chooses Adam algorithm as the optimizer, which performs

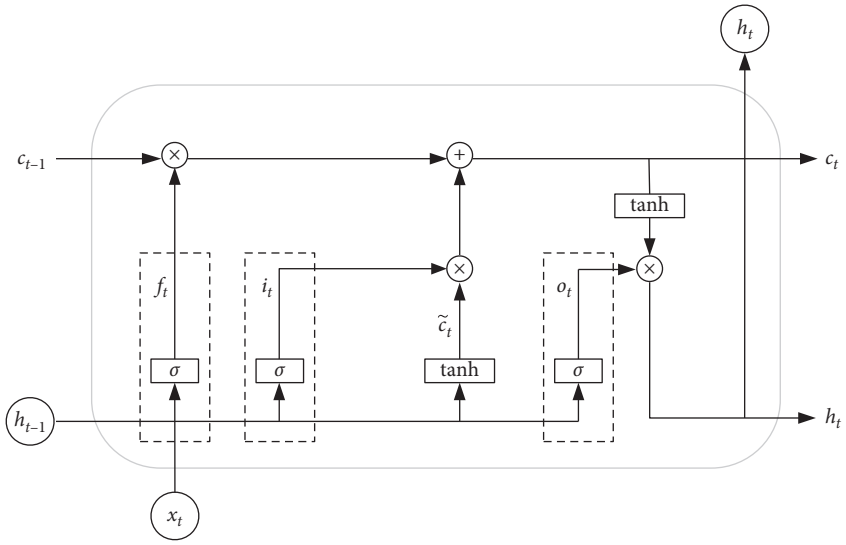


Figure 2: The basic of LSTM memory cell.

better in practice than other stochastic optimization methods based on gradient.

4. The Proposed BiLSTM Deep Learning Model

The situation of satellite operation is characterized by telemetry data transmitted from satellite, which has a certain spatial and temporal correlation. BiLSTM can not only solve the problem of long-term dependence between current moment and past moment, but also pay attention to the correlation between current moment and future moment. Based on this, the paper introduces BiLSTM to predict the operation trend of satellite. But, in practical application, we cannot get the future time series data, so that the paper presents using LSTM model to predict the future trend and takes the prediction results and historical data as the input of BiLSTM. Although BiLSTM is a deep neural network in time sequence, its network structure is extremely shallow in the number of layers. Increasing the number of depth layers of neural network can increase the processing capacity of neural network. The paper proposes stacking multiple BiLSTM and designing a deep bidirectional long short-term memory neural network. By stacking multiple hidden states, the deep BiLSTM is expanded to predict the operation trend of meteorological satellite. Trend prediction framework is shown in Figure 3.

The Deep BiLSTM network model contains multiple hidden layers, and the output of the former layer is used as the input of the latter layer, which helps mine the relationship between the front and back time series data, enrich the feature set of network learning, and improve the accuracy of the model. In the model, the transmission of network parameters in the input layer is the same as BiLSTM. But, in the out layer, Deep BiLSTM transmits the state to the hidden layer as input, and the state of multiple hidden layers is transmitted upward layer by layer, and finally to the output layer, as shown in Figure 4.

The historical time series data is input as the forward data of Deep BiLSTM network model, and the future time series feature data predicted by LSTM is input as the backward data of Deep BiLSTM network model.

In BiLSTM network, the historical time series data $X = [x_1, x_2, \ldots x_T]$ is input to the forward network unit of BiLSTM, which gets the forward hidden layer state $\overrightarrow{h}_t$. Historical time series data $X = [x_1, x_2, \ldots x_T]$as the input of LSTM network, is used to predict the future time series data $X = [x'_1, x'_2, \ldots x'_T]$. The predicted time series data $X = [x'_1, x'_2, \ldots x'_T]$ is input to the backward network unit of BiLSTM, which gets the backward hidden layer state $\overleftarrow{h}_t$. The output p_t is obtained by $\overrightarrow{h}_t$and$\overleftarrow{h}_t$, which are expressed as follows:

$$\overrightarrow{h}_t = \sigma\left(W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right), \tag{8}$$

$$\overleftarrow{h}_t = \sigma\left(W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right), \tag{9}$$

where σ is the nonlinear activation function of the hidden layer, $W_{x\overrightarrow{h}}$ is the weight from the input x of the current neuron to the hidden layer $\overrightarrow{h}_t$ at this moment, $W_{\overrightarrow{h}\overrightarrow{h}}$ is the weight from the state quantity at the previous moment to the current state quantity, $\overrightarrow{h}_{t-1}$ is the output value of the hidden layer at the previous moment, and $b_{\overrightarrow{h}}$ is the offset term.

The BiLSTM model is iterated in two directions at the same time and calculates the predicted value p_t by weighting the hidden layer state, which is expressed as

$$p_t = W_{\overrightarrow{h}p} \overrightarrow{h}_t + W_{\overleftarrow{h}p} \overleftarrow{h}_t + b_p. \tag{10}$$

Here, $W_{\overrightarrow{h}p}$ is the weight from the hidden layer $\overrightarrow{h}_t$ to the output, $W_{\overleftarrow{h}p}$ is the weight from the hidden layer $\overleftarrow{h}_t$ to the output, and b_p is the offset term.

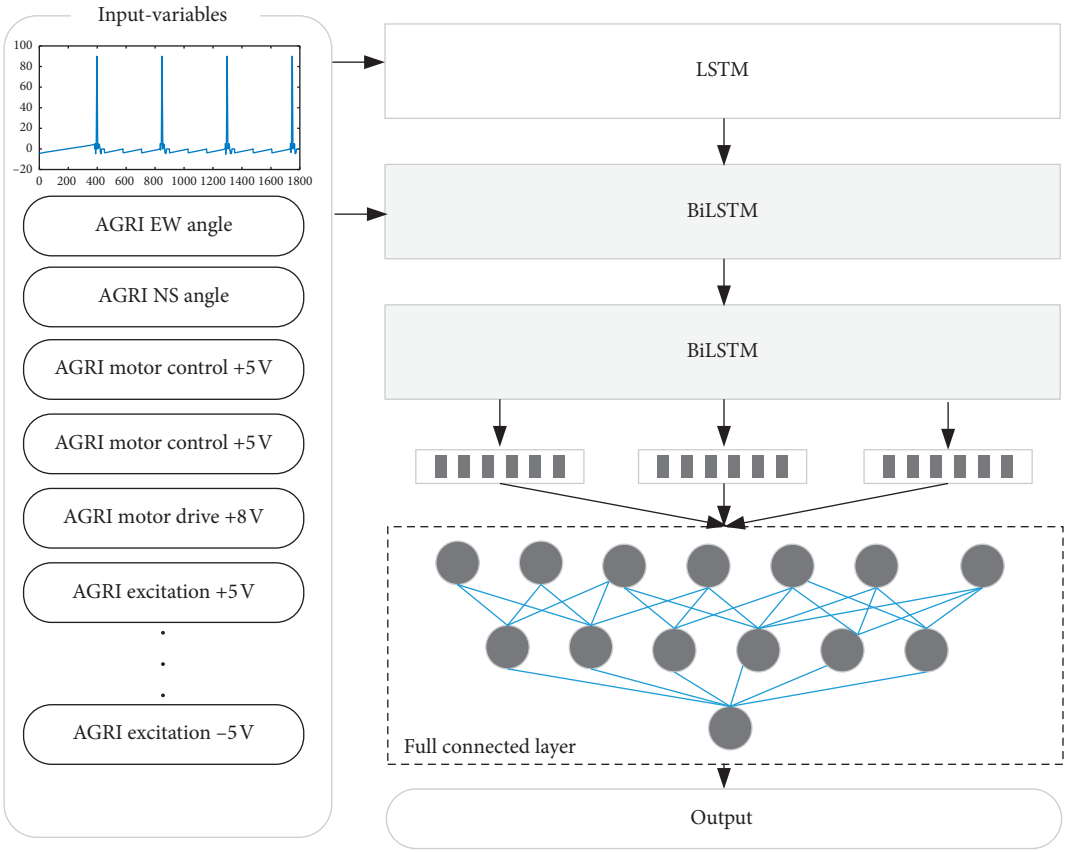


Figure 3: Trend prediction framework for meteorological satellite.

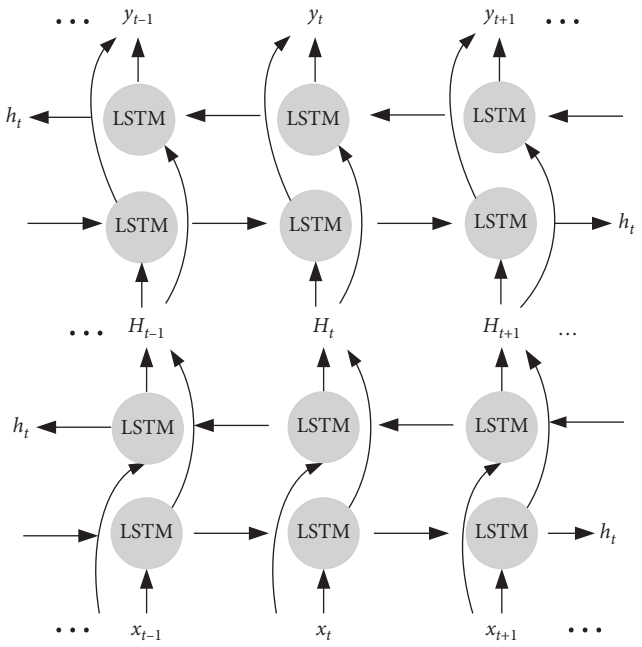


Figure 4: The network structure of BiLSTM.

5. Experiments

5.1. Datasets. FY4A on orbit is the second generation of geostationary meteorological satellite. It has four loads, that is, Advanced Geostationary Radiation Imager (AGRI), Geostationary Interferometric Infrared Sounder (GIIR), Lightning Mapping Imager (LMI), and Space Environment Monitoring Instrument Package (SEP). Among these, the operation trend of AGRI is used to evaluate the performance of the approach proposed in the paper, whose main function is to realize the quantitative observation of Earth's surface and clouds. The telemetry data of AGRI, which are generated and transmitted on certain days, are put into use for identifying the algorithm.

Among telemetry data, 26 observation variables most likely to be relevant to the operation trend, that is, 26 features, are selected. Therefore, each sample data for trend prediction consists of 26 variables, that is, $X = [TMC1, TMC2, \ldots, TMC26]$, which are shown in Table 1. The dataset is generated at a sampling interval of 2 second. A total of 43189 data samples are collected as training samples and test samples. Figure 5 shows that the variables change with time from UTC 01 : 00 to UTC 01 : 59 on February 23, 2019. Figures 5(a)-5(d) are, respectively, the changing situation of variable *TMC3*, *TMC6*, *TMC11*, and *TMC19*with time. From Figure 5, we can see that different variables can have different operation trends with time.

6. Evaluation Metric

In order to evaluate the prediction effect of BiLSTM deep learning model proposed by this paper, the Mean Absolute

Table 1: Variables of satellite telemetry.

Variable name	Variable symbol
AGRI scanning mechanism status	*TMC*1
AGRI EW angle	*TMC*2
AGRI NS angle	*TMC*3
AGRI 0.55 preamplifier +5 V	*TMC*4
AGRI 0.45 preamplifier +5 V	*TMC*5
AGRI 1.36 preamplifier +5 V	*TMC*6
AGRI 0.55 preamplifier bias voltage	*TMC*7
AGRI 0.45 preamplifier bias voltage	*TMC*8
AGRI 1.36 preamplifier bias voltage	*TMC*9
AGRI visible/near-infrared main amplifier +5 V	*TMC*10
AGRI visible/near-infrared main amplifier −5 V	*TMC*11
AGRI infrared main amplifier +12 V	*TMC*12
AGRI infrared main amplifier −12 V	*TMC*13
AGRI medium-long wave photoconduction preamplifier +12 V	*TMC*14
AGRI medium-long wave photoconduction preamplifier −12 V	*TMC*15
AGRI medium wave photovoltaic preamplifier +1.5 V	*TMC*16
AGRI medium wave photovoltaic preamplifier −1.5 V	*TMC*17
AGRI motor control +5 V	*TMC*18
AGRI motor drive +8 V	*TMC*19
AGRI temperature control +5 V	*TMC*20
AGRI excitation +5 V	*TMC*21
AGRI excitation −5 V	*TMC*22
AGRI induct synchronizer +15 V	*TMC*23
AGRI induct synchronizer −15 V	*TMC*24
AGRI simulation conditioning +5 V	*TMC*25
AGRI simulation conditioning −5 V	*TMC*26

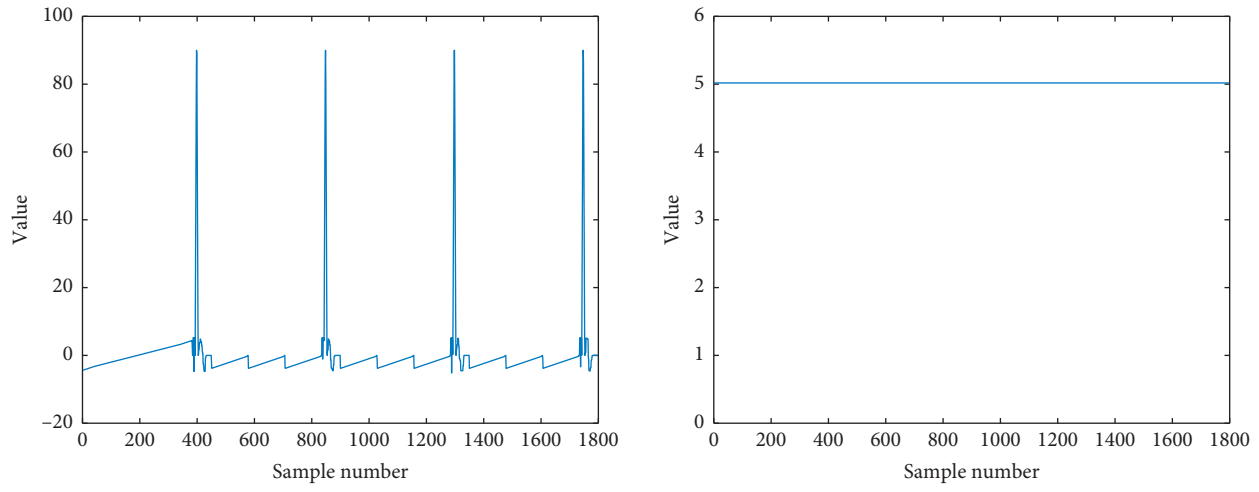


Figure 5: Continued.

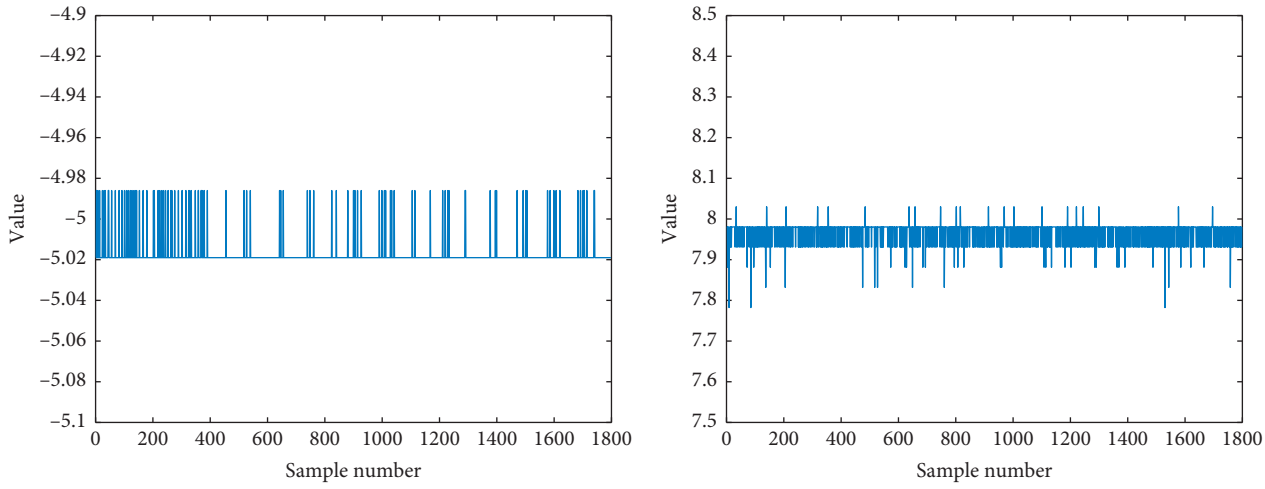


Figure 5: The variables changing with time. (a) TMC3 variable. (b) TMC6 variable. (c) TMC11 variable. (d) TMC19 variable.

Error (MAE) and Root Mean Square Error (RMSE) are selected as the criteria.

The MAE calculation formula is as follows:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{y}_i - y_i\right|, \tag{11}$$

where $\hat{y}_i$ and y_i represent the predicted value and the actual value, respectively, and n is the number of samples. The smaller the MAE is, the more accurate the prediction is.

The RMSE calculation formula is as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2}, \tag{12}$$

where n is the number of samples, $\hat{y}_i$ is the predicted value, and y_i is the actual value. The smaller the RMSE is, the more accurate the prediction is.

MAE and RMSE both can measure the error of the predictions. Their difference is that RMSE can penalize huge errors, but MAE cannot.

7. Results and Discussion

In order to prove the effectiveness of the BiLSTM deep learning model proposed by the paper, this method is compared with LSTM using the same training set and test set data under the same operating environment. All the experiments are carried out under the running environment of Inter(R) Core(TM) i7- 1.99 GHz, 8.00 GB RAM, and Windows 10. The development environment is Python 3.6 along with TensorFlow 2.0.0.

For the sake of reducing the dimension difference of characteristics and its possible impact, all data are normalized. After data preprocessing, roulette method is used to divide data sets, and 70% of the data set is used as training set, 20% as verification set, and the remaining 10% as test set. Because the setting of experimental parameters has a great influence on the experimental results, the method of fixed parameters is used in the experiment. Multiple experiments are done when the dropout is 0.1, 0.2, . . ., 0.9, respectively. Among them, when dropout value is 0.5, the RMSE is the smallest; that is, the prediction accuracy is the highest, as shown in Figure 6. By comparing the results of many experiments, it is found that the prediction effect of BiLSTM deep learning model proposed is the best when the parameters in Table 2 are taken. The epoch is 100, the optimizer chooses Adam, batch size is 64, training steps are 200, and learning rate is 0.0001.

Considering that the depth of neural network can improve the model representation, the number of BiLSTM layers is set to two. The BiLSTM layer number is determined by evaluating the loss value and the prediction of different BiLSTM layers, which are, respectively, shown in Figure 7 and Table 3. Figure 7 illustrates that the loss value of two layers is smaller than that of one layer. In Table 3, the RMSE and MAE of two layers are 0.0231 and 0.0188, respectively, which are smaller than those of one layer. These mean that the two layers are superior to one layer. Thus, the layer number of the BiLSTM model proposed by the paper is chosen as two.

In the experiment, infrared main amplifier −12 V, medium wave photovoltaic preamplifier +1.5 V, medium wave photovoltaic preamplifier −1.5 V, motor control +5 V, motor drive +8 V, temperature control +5 V, excitation +5 V, excitation −5 V, induct synchronizer +15 V, induct synchronizer −15 V, simulation conditioning +5 V, and simulation conditioning −5 V that sign the operation situation of meteorological satellite are predicted, which are labeled as $R1$, $R2 \ldots R12$, respectively. The training set data is used to train BiLSTM deep learning model, LSTM model, and RNN model. The verification set data is applied to adjust parameter of the model. And the model achieved by training is used to predict the test set data. The evaluation error indexes of each method can be calculated on the base of the predicted value of each method and the real value. The comparison results of the three

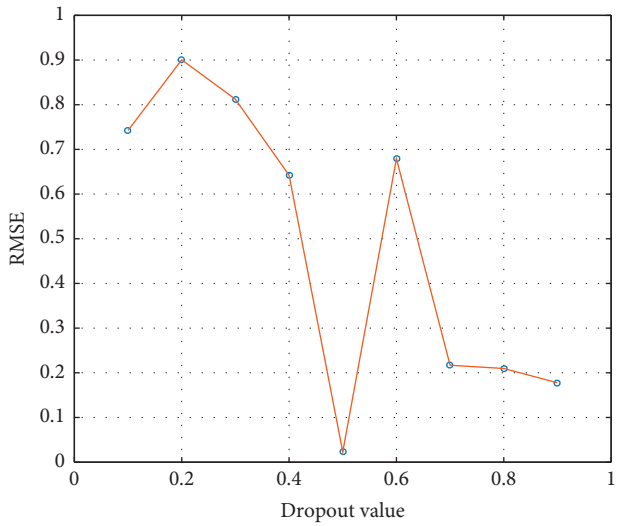


Figure 6: Dropout curve.

Table 2: Parameter settings of BiLSTM model.

Parameter	Value	Parameter	Value
Epoch	100	Batch size	64
Optimizer	Adam	Learning rate	0.0001
Training steps	200	Dropout	0.5

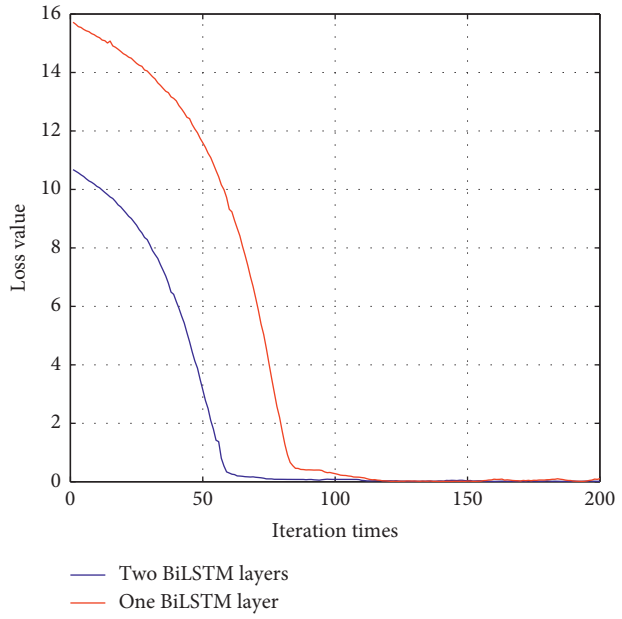


Figure 7: Loss value for different number of BiLSTM layers.

Table 3: Prediction error for different number of BiLSTM layers.

	One BiLSTM layer	Two BiLSTM layers
RMSE	0.3276	0.0231
MAE	0.3259	0.0188

methods are shown in Table 4. We can see that the RMSE of BiLSTM is smaller than LSTM and RNN from Table 4. This means that the BiLSTM deep learning model proposed in this paper can better predict the operation trend of meteorological satellite than LSTM and RNN models.

Table 4: Various error indicators for prediction using different model.

	RMSE (BiLSTM)	RMSE (LSTM)	RMSE (RNN)
*R*1	0.02213	0.25671	0.69161
*R*2	1.14932	3.65821	2.07490
*R*3	1.13257	2.98576	2.22344
*R*4	0.03683	0.67384	1.19673
*R*5	0.03285	0.43784	2.14687
*R*6	0.06222	0.98673	0.63541
*R*7	0.39643	1.25473	0.39750
*R*8	0.02880	0.36528	1.69788
*R*9	0.01315	0.19327	0.21264
*R*10	1.13424	3.06785	1.99287
*R*11	0.03279	0.42356	1.07320
*R*12	0.02310	0.27856	0.65139

8. Conclusions

In order to avoid the occurrence of major satellite fault and provide the complete and accuracy satellite data, this paper proposes meteorological satellite operation prediction method. Considering the unstable, nonlinear, and temporal characteristics of satellite telemetry data, a BiLSTM deep learning model for satellite operation trend prediction is put forward. The method uses multidimensional telemetry variables as the input, and LSTM is used to predict the future trend, and then, we take the prediction results and historical data as the input of BiLSTM deep learning model. Increasing the number of depth layers of neural network can enrich the feature set of network learning, increase the processing capacity of neural network, and improve the accuracy of the model. The research on the data sets transmitted from AGRI load of FY4A satellite shows that the BiLSTM deep learning model proposed by the paper has highest prediction accuracy and better performance than the classical LSTM model.

Further research work will be undertaken to improve the current method to make the results more accurate based on more dataset. Further work will also research that the method can be applied with real-time prediction for satellite, rather than just focusing on existing historical data set.

Data Availability

The telemetry data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was supported by the National Key Research and Development Program of China (No.2018YEC1507803).

References

[1] A. Al-Smadi, "The estimation of the order of an ARMA process using third-order statistics," *International Journal of Systems Science*, vol. 36, no. 15, pp. 975–982, 2005.

[2] A. Kizilkaya and A. H. Kayran, "ARMA model parameter estimation based on the equivalent MA approach," *Digital Signal Processing*, vol. 16, no. 6, pp. 670–681, 2006.

[3] F. Yuan, U. Kumar, and D. Galar, "Reliability prediction using support vector regression," *International Journal of System Assurance Engineering and Management*, vol. 263, 2010.

[4] S. Safarzadegan Gilan, H. Bahrami Jovein, and A. A. Ramezanianpour, "Hybrid support vector regression - particle swarm optimization for prediction of compressive strength and RCPT of concretes containing metakaolin," *Construction and Building Materials*, vol. 34, pp. 321–329, 2012.

[5] K. F. Fung, Y. F. Huang, and C. H. Koo, "Coupling fuzzy-SVR and boosting-SVR models with wavelet decomposition for meteorological drought prediction," *Environment Earth Science*, vol. 78, 2019.

[6] W. Gao and J. Han, "Prediction of destroyed floor depth based on principal component analysis (PCA)-Genetic algorithm (GA)-Support vector regression (SVR)," *Geotechnical and Geological Engineering*, vol. 38, no. 4, pp. 3481–3491, 2020.

[7] G. Hao, "Study on prediction of urbanization level based on GA-BP neural network," in *Proceedings of the 21st International Conference on Industrial Engineering and Engineering Management*, pp. 1021–1026, Singapore, 2019.

[8] J. D. Watson and F. H. C. Crick, "BP neural network-based product quality risk prediction," *Advances in Intelligent Systems and Computing*, vol. 1117, pp. 1021–1026, 2019.

[9] F. He and L. Zhang, "Mold breakout prediction in slab continuous casting based on combined method of GA-BP neural network and logic rules," *The International Journal of Advanced Manufacturing Technology*, vol. 95, pp. 4081–4089, 2018.

[10] C. Yuan, D. Niu, C. Li, L. Sun, and L. Xu, "Electricity consumption prediction model based on bayesian regularized BP neural network," *Advances in Intelligent Systems and Computing*, vol. 928, pp. 528–535, 2020.

[11] N. R. Metu and T. Sasikala, "Prediction analysis on web traffic data using time series modeling, RNN and ensembling techniques," *Lecture Notes on Data Engineering and Communications Technologies*, vol. 26, pp. 611–618, 2018.

[12] C. Lin and M. Chi, "A comparisons of BKT, RNN and LSTM for learning gain prediction," *Lecture Notes in Computer Science*, vol. 10331, pp. 536–539, 2017.

[13] Q. Wu, K. Ding, and B. Huang, "Approach for fault prognosis using recurrent neural network," *Journal of Intelligent Manufacturing*, vol. 31, no. 7, pp. 1621–1633, 2020.

[14] N. K. Manaswi, "RNN and LSTM," in *Deep Learning with Applications Using Python,* Apress, Berkeley, CA, USA, 2018.

[15] S. Poornima and M. Pushpalatha, "Drought prediction based on SPI and SPEI with varying timescales using LSTM recurrent neural network," *Soft Computing*, vol. 23, no. 18, pp. 8399–8412, 2019.

[16] F. Mtibaa, K.-K. Nguyen, M. Azam, A. Papachristou, J.-S. Venne, and M. Cheriet, "LSTM-based indoor air temperature prediction framework for HVAC systems in smart buildings," *Neural Computing and Applications*, vol. 32, no. 23, pp. 17569–17585, 2020.

[17] J. Zhao, J. J. Wu, X. W. Guo et al., "Prediction of radar sea clutter based on LSTM," *Journal of Ambient Intelligence and Humanized Computing*, 2019.

[18] V. Gundu and S. P. Simon, "PSO-LSTM for short term forecast of heterogeneous time series electricity price signals," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 2, pp. 2375–2385, 2021.

[19] S. N. Sima, T. Neda, and S. N. Akbar, "The performance of LSTM and BiLSTM in forecasting time series," in *Proceedings of the IEEE International Conference on Big Data*, Los Angeles, CA, USA, 2019.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] W. J. Lu, J. Z. Li, J. Y. Wang et al., "A CNN-BiLSTM-AM method for stock price prediction," *Neural Computing and Applications*, vol. 33, pp. 4741–4753, 2020.

[22] A. Graves, "Long short-term memory," *Studies in Computational Intelligence*, vol. 385, pp. 37–45, 2012.

[23] S.-I. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[24] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 7, pp. 257–269, 2011.

[25] S. Yeung, O. Russakovsky, N. Andriluka, G. Mori, and L. Fei-Fei, "Every moment counts: dense detailed labeling of actions in complex videos," *International Journal of Computer Vision*, vol. 126, no. 2-4, pp. 375–389, 2018.

[26] D. P. Kingma and L. J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the International Conference On Learning Representations*, (ICLR), San Diego, CA, USA, 2015.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6658066, 10 pages
https://doi.org/10.1155/2021/6658066

Research Article

Cache Pollution Detection Method Based on GBDT in Information-Centric Network

Dapeng Man, Yongjia Mu, Jiafei Guo, Wu Yang, Jiguang Lv, and Wei Wang

Information Security Research Center, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Wei Wang; w_wei@hrbeu.edu.cn

Received 25 December 2020; Accepted 7 June 2021; Published 16 June 2021

Academic Editor: M.A. Jabbar

There is a new cache pollution attack in the information-centric network (ICN), which fills the router cache by sending a large number of requests for nonpopular content. This attack will severely reduce the router cache hit rate. Therefore, the detection of cache pollution attacks is also an urgent problem in the current information center network. In the existing research on the problem of cache pollution detection, most of the methods of manually setting the threshold are used for cache pollution detection. The accuracy of the detection result depends on the threshold setting, and the adaptability to different network environments is weak. In order to improve the accuracy of cache pollution detection and adaptability to different network environments, this paper proposes a detection algorithm based on gradient boost decision tree (GBDT), which can obtain cache pollution detection through model learning. *Method.* In feature selection, the algorithm uses two features based on node status and path information as model input, which improves the accuracy of the method. This paper proves the improvement of the detection accuracy of this method through comparative experiments.

1. Introduction

With the popularity of the Internet and Internet of Things technologies and the transformation of IPV4 and IPV6 technology, more and more smart devices can access the Internet, and network traffic is beginning to grow rapidly. The current application range and scale of the Internet have far exceeded the original intention of the design. The information-centric network [1–4] was proposed, hereinafter referred to as ICN. The current information-centric network has protocols such as NDN [5], PSIRP [6], DONA [7], and NetInf [8]. Although these protocols have different forms, the key point is to use content names or IDs to obtain content, and all routers that pass through are supported for caching. Among the many information-centric network implementations, the most mainstream feasible solution is the named data network, hereinafter referred to as NDN. All the solutions discussed in this paper are based on NDN.

Since the original intention of the information-centric network design includes the use of caching to increase network utilization, the cache is an indispensable part of the information-centric network. If there is no cache, the efficiency of the network will be significantly reduced. In IP-based networks, there are various kinds of network attacks, and one of the well-known attacks is the DDoS attack [9, 10]. Unlike the IP network, the principal part of the information-centric network is content rather than IP. The attacker cannot specify a certain packet to send to the target host. Therefore, the information-centric network has inherently capable resistance to such attacks. However, due to the massive use of cache in the information-centric network to improve network efficiency, it naturally brings the cache pollution attack. The attacker can send many nonpopular content requests through the controlled host so that the routers on the path cache the nonpopular content. When the normal user makes a request, the cache hits failed because the node cache cannot find the corresponding content, the router only forwards the request to the content producer for processing, which makes the original intention of the information-centric network design, using the cache to optimize the network message to the maximum extent becomes useless, so that the traffic of the backbone link of the network

is greatly increased, resulting in network congestion and other phenomena.

Although ICN has rethought some of the design concepts of optimization and innovation, in many respects, some central issues have not been completely resolved in the initial ICN network framework. This paper mainly discusses the problem of cache pollution detection. Cache pollution attack is one of the most serious attacks in information-centric networks. Most of the current detection algorithms need to manually set thresholds. These methods have poor adaptabilities to different environments. Therefore, in this paper, a GBDT-based cache pollution detection method is proposed, which does not require a manual setting of thresholds and has high accuracy.

1.1. Related Work. NDN is an information-centric network architecture with high scientific research value and development potential [5, 11], which has attracted great attention from the academic community in recent years. Although the native architecture of NDN tries to provide certain data security by encrypting and signing data packets by network content producers, in the face of various malicious attacks under complex environmental conditions in the actual network, its network nodes still have great potential risks, for example, Distributed Denial-of-Service (DDoS) attacks and cache pollution. Research by Virgilio et al. [12] shows that DDoS attacks can use a large number of forged interests to exhaust the memory of the pending interest table (PIT) in the NDN node. Tan Nguyen et al. calculated the difference in the satisfaction rate of interest packets caused by DDoS to detect and defend DDoS [13], but the effect of this solution is not ideal in the face of network cache pollution attacks.

Cache pollution attacks [14] have been widely studied in IP-based networks. More research studies focus on the cache of web traffic. The current research divides cache pollution attacks into two categories, destroying the content distribution (locality-disruption) characteristics attack and the false-locality feature attack [15]. In the destruction of the content distribution model, the attacker or the controlled host sends many interest packets of nonpopular content to the network. Therefore, the router' CS table in the network caches is occupied by a large amount of nonpopular content, so that the requests of the normal popular contents cannot utilize the CS table of the router, increasing the network delay and achieving the purpose of the attack; in the forged content distribution attack, the attacker's attack model, and the broken content distribution differently, the attack does not destroy the overall distribution feature of the content in the network, but periodically sends a request for non-streaming content, so that the cache is occupied by non-popular contents for a long time, resulting in a decrease in network cache capability.

Although cache pollution attacks have been extensively studied in IP networks, most of the detection methods for cache pollution in IP networks cannot be applied to ICN networks. This is because in ICN networks, the request packet does not contain any information of the requester, so it cannot be the source of the request packet which is traced, and the attack object of the cache pollution in the ICN is also different from that in the IP network. The cache pollution attack in the IP network is mostly directed to the cache server, and the target of the cache pollution attack in the ICN is the routing node in the network. This also shows that, in ICN cache pollution attacks, cache routes are difficult to perceive the existence of attacks [16, 17]. As an information-centric network architecture with network cache, NDN is vulnerable to cache poisoning and pollution attacks.

Mauri et al. consider a case in which an attacker uses NDN's routing and caching system to add a large amount of malicious content to the cache storage of network nodes [18]. Literature [19] analyzed the essential characteristics of content pollution attacks in NDN networks based on some cases. Li et al. proposed a lightweight integrity verification and access control mechanism for network cache pollution attacks [20]. Literature [21] studied local cache pollution attacks and proposed a cache shield, which enhances the robustness of the network by increasing the cost of cache pollution attacks.

Park et al. proposed a matrix random check method [22], which uses the content name. The method is mapped to the matrix. Each position of the matrix represents the number of corresponding requests. Whenever a request arrives, the rank of the matrix is checked. When the rank of the matrix is below a certain threshold, this node is considered to be attacked. Conti et al. proposed a lightweight mechanism [23] for detecting cache pollution attacks, which first defines a random sample set of content, monitors the distribution of current sample sets, and dynamically counts the arrival rates of these requests, once these cache pollutions have occurred when the arrival rate changes and exceeds a certain threshold.

2. Analytical Methods

Caching is one of the reasons why information-centric networks are efficient. This chapter will discuss the cache pollution detection problem in the information-centric network and study how to use the machine learning method to solve the problem that the traditional detection model needs to manually set the threshold. Then, a GBDT-based cache pollution detection model is proposed, which is shown in Figure 1. First, the node status information and path information are collected as features, and then the gradient boosting tree algorithm is combined with the two features to build a cache pollution detection model. The training speed of this model is fast, and the accuracy is high. Finally, we evaluate the model through experiments.

2.1. GBDT Model. The cache pollution detection model is essentially a classifier and is a two-class classifier. One is that the current node is being attacked, and the other is that the current node is not attacked. This chapter uses the GBDT model for cache pollution detection. GBDT is the abbreviation of gradient boost decision tree, which is the gradient lifting tree. This model is practically a gradient promotion model in the decision tree, that is, multiple decision trees are

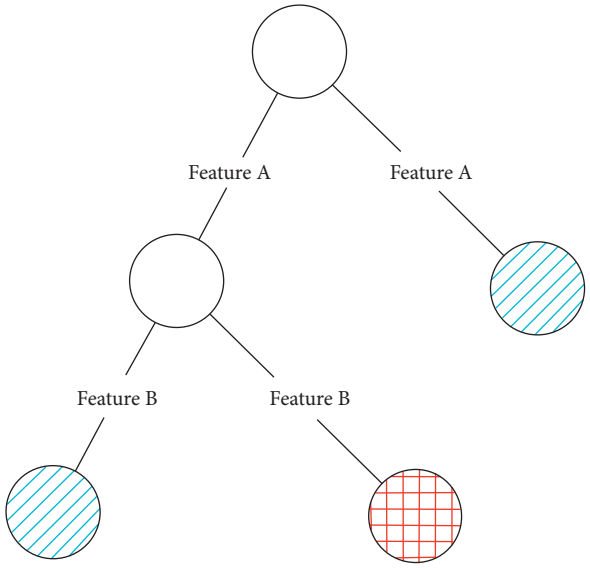


Figure 1: Schematic diagram of the decision tree.

fused according to the gradient promotion method. This section will introduce the model-related content .

(1) The decision tree is divided into a classification tree and a regression tree. The classification tree refers to the decision tree used for classification problems. This chapter uses the GBDT model to classify the cache pollution problem. Therefore, only the classification tree is introduced here. The classification tree is a kind of classification model using the tree structure. Each leaf node of the tree represents the classification result. Each branch of the tree is a decision mode. As shown in Figure 2, according to a set of new features A and B and the trunk of each tree according to the feature selection, the child node of the tree is entered. If the child node is a leaf node, the classification result is obtained. Therefore, for any training set, if there is no data with the same characteristics but different results, the decision tree can obtain 100% accuracy on the training set. This is because the depth of the tree can grow all the time, and the decision algorithm will eventually get the result after judging all the attributes. Although the decision tree can get almost 100% accuracy on the training set, the test set may get very poor results. This is because the decision tree is just remembered all the training data, but did not learn how to judge this classification problem which the data that does not appear in the training set. Thus, it will produce more random results. This phenomenon is also called overfitting problem.

For the decision tree, its training process aims to determine the characteristics of each branch, that is, finding the optimal feature in each node state. The usual method is to use information gain and information gain ratio and Gini index.

The formula for information entropy is defined as follows:

$$H(U) = E[-\log\ p_i] = -\sum_{i=1}^{n} p_i \log\ p_i. \tag{1}$$

To facilitate the operation, the log here takes the base 2 logarithm. From the formula, the value range of $H(U)$ is [0, 1], and the amount of information entropy quantitatively identifies the uncertainty of the information. A larger value indicates a stronger uncertainty, that is, less information is included. For example, a box has red and white balls. If there are no restrictions, you can only assume that the probability distribution is 1/2 and 1/2. Then, the entropy now is 1, which is the maximum value. This indicates that the information of the situation is the least, and if we know that there is only a white ball in this box, the probability distribution becomes 0 and 1. In this event, the information entropy is 0, which means that the amount of information is the largest currently, without any uncertainty.

The ID3 algorithm uses the information gain method, that is, the change of the information entropy before and after the split is used to measure the optimal attribute, and the information gain obtained by dividing the D state by the feature A is defined as equation (2), that is, each iteration selects the largest information gain value to produce subsets of the data.

$$\text{Gain}(D, A) = \text{Entropy}(D) - \text{Entropy}(D, A). \tag{2}$$

However, the ID3 algorithm has a significant limitation. The ID3 algorithm will choose the minimal Entropy(D, A), which will lead to the algorithm tending to select those features with more subclasses and purer features; therefore, the C4.5 algorithm is proposed to solve the ID3' s drawback. The C4.5 algorithm uses the information gain rate as the measure of the best feature. The optimal gain rate Gain_Rate(D, A) is defined as equation (3), that is, the attribute with the maximum normalized information gain is chosen to make the decision.

$$\text{Gain\_Rate}(D, A) = \frac{\text{Entropy}(D)}{\text{Entropy}(D, A)}. \tag{3}$$

The CART tree can be used for both classification and regression problems. The CART tree uses the Gini index or Gini impurity to determine the optimal division point. The Gini index is defined as equation (4), and the Gini index can also represent the uncertainty of the sample set S. Since the cart is a binary decision tree, each partition can only divide the set into two parts, so each partition needs to use the i th attribute value of attribute A, as shown in equation (5). $\text{Gain}_{A,i}(S)$ represents the uncertainty of set s after A_i segmentation. The larger the Gini index, the greater the uncertainty of the sample set S after A_i division, which is like entropy.

$$\mathrm{GINI}(S) = 1 - \sum P_k^2, \tag{4}$$

$$\mathrm{Gain}_{A,i}(S) = \frac{n1}{N}\mathrm{GINI}(S_1) + \frac{n2}{N}\mathrm{GINI}(S_2). \tag{5}$$

(2) The boosting algorithm [24] is an important part of ensemble learning. Boosting can be used to primarily reduce the bias of the model and enhance the weak models. The principle of the boosting algorithm is to use a weak model to fuse the strong model. First, weak classifier α is trained with the initial weights. The data weight of the training set is updated according to the prediction results of the weak classifier α so that the weight of the sample points whose prediction is error was made by the weak classifier α is increased. Therefore, these samples with a high error rate can get higher accuracy in later learning with weak learner β. Then, we use the weighted training set to train weak classifier β, which is repeated until the number of weak classifiers reaches the number T appointed beforehand. Finally, these T weak classifiers are assembled through a set strategy to obtain the final strong classifier.

(3) GBDT is one of the ensembles learning boosting algorithm. GBDT is also an iterative model that uses a forward distribution algorithm, but the weak learner limits the use of the CART regression tree model.

For the cache pollution problem, binary classification is needed. The log loss function can be used, and the loss function is shown as follows:

$$L(y, f(x)) = \log\left(1 + \exp(-yf(x))\right). \quad (y \in (-1, +1)) \tag{6}$$

The negative gradient of the loss function of the ith sample of the tth round is expressed as follows:

$$r_{\mathrm{ti}} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f_{t-1}(x)}. \tag{7}$$

Here, the loss function of the cache pollution problem is brought in, and the negative gradient error at this time is as follows:

$$r_{ti} = \frac{y_i}{\left(1 + \exp\left(y_i f(x_i)\right)\right)}. \tag{8}$$

Using (x_i, r_{ti}) $(i = 1, 2, \ldots, m)$, we can fit a CART regression tree and get the tth regression tree, and its corresponded leaf node region R_{tj}, $j = 1, 2, \ldots, J$, where J is the number of leaf nodes.

For each sample in the leaf node, the loss function is minimized, and the best output value c_{tj} of the fitting leaf node is as follows.

$$c_{tj} = \underbrace{\arg\min}_{c} \sum_{x_i \in R_{tj}} L\left(y_i, f_{t-1}(x_i) + c\right). \tag{9}$$

For the problems provided in this chapter, the loss function of the cache pollution problem is brought in, and the optimal residual fit value of each leaf node is as shown in equation (10).

$$c_{tj} = \underbrace{\arg\min}_{c} \sum_{x_i \in R_{tj}} \log\left(1 + \exp\left(-y_i\left(f_{t-1}(x_i) + c\right)\right)\right). \tag{10}$$

Since the above formula is more difficult to optimize, we use an approximation instead, as follows:

$$c_{tj} = \frac{\sum_{x_i \in R_{tj}} r_{tj}}{\sum_{x_i \in R_{tj}} \left|r_{tj}\right|\left(1 - \left|r_{tj}\right|\right)}. \tag{11}$$

Thus, the fitting function for each iteration is obtained as follows:

$$h_t(x) = \sum_{j=1}^{J} c_{tj} I, \quad \left(x \in R_{tj}\right). \tag{12}$$

Finally, the resulting strong learner expression is given as follows:

$$f_t(x) = f_{t-1}(x) + \sum_{j=1}^{J} c_{tj} I, \quad \left(x \in R_{tj}\right). \tag{13}$$

2.1.1. Node Status Information. In the NDN-based network and the IP-based network, the cache pollution attack has similarities. In both networks, the attacker attempts to attack a terminal. In the IP-based network, the terminal refers to some servers. In the network of the NDN architecture, this terminal is a certain router, so this kind of attack is an attack that only the attacked node can judge.

In the NDN, the most intuitive reflection of the attack that occurred is the cache hit rate of the normal request, but the intermediate router responsible for forwarding and caching cannot distinguish the difference between the normal request interest packet and the attack interest packet, so the data cannot be directly or indirectly obtained through the router. It is only possible to estimate whether an attack has occurred by some available quantities. The variable quantities available in the NDN router are shown in Table 1.

Since the NDN is designed to follow the "thin waist" principle, it has less available information for routers. Firstly, the cache pollution attack is implemented by sending abundant nonpopular interest packets to the network. Therefore, the correlation amount of the data packet does not have much significance and is not a feature. Secondly, for the attack detection model, some overall quantities have no meaning of the model detection, such as the total number of interest packets and the total cache hit rate, so these quantities are not suitable as model parameters. In addition, some ID-type quantities such as the name of the interest package and the cached interest package name are substantially independent of the cache attack. Thus, such ID-type variables should not be used as the features of the model. Existing research has shown that, for interest packet requests in routers, the Zip-f distribution is normally satisfied, that is, the most frequent requests are only a small part

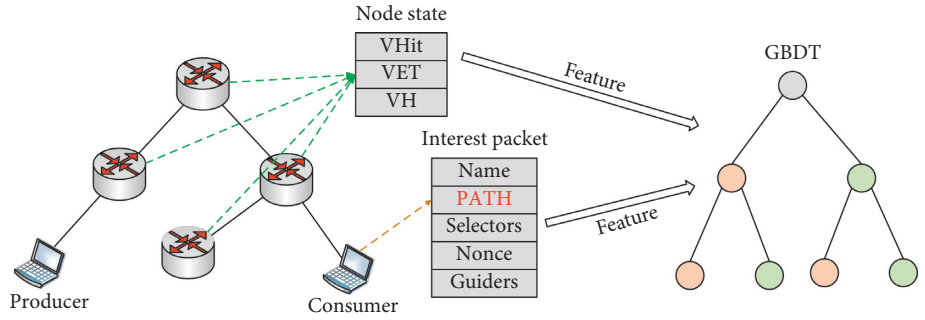


Figure 2: Schematic diagram of GBDT cache contamination detection.

Table 1: NDN router availability.

Available in NDN routers	Description
Interest package content name, IName	Obtained from the content name field of the interest package
The content name of the packet, DName	Obtained from the content name field of the data packet
Cached content name, CName	Get the content name from the CS table
Cache replacement rate, pmiss	Replacement rate by calculation
Cache hit ratio, EH	Cache hit rate by calculation
Number of interest packages, TI	Accumulate the number of interest packages
Number of data packets, TD	Accumulate the number of data packets

of all the data [25]. Therefore, when retrieving features, we should consider extracting features that can reflect the distribution of content. Considering that the number of interest packets per unit time can reflect the distribution of content, the number of interest packets with the largest number of first K requests per unit time is used to form a K-dimension feature to enable the model to learn the current distribution characteristics and then select the K-cache hit rate of corresponding content as the feature.

For the above features, the number of interest packets may greatly depend on the usage of the network. For example, the total number of interest packets in the high-interval and low-peak networks varies greatly, but the difference does not mean whether it is attacked. Therefore, if you directly select the number of interest packages as a feature, the model may be too dependent on the number of packets in the network. Therefore, it is necessary to normalize the number of interest packets, not using the quantity, using the proportion as a feature, the normalization formula is as shown in equation (14), and cnt_k indicates the number of K interest packages with the largest number of units of interest per unit time, total represents the total number of interest packets per unit time, and the characteristics of the final selection node are shown in Table 2. In Table 2, VHit is cache replacement rate under cache replacement policy which can be obtained by the source code of ndnSIM because we use the built-in cache policy, that is, the LRU policy. And, VH is the K cache hit ratios corresponding to interest packages also can be obtained by the source of ndnSIM.

$$\text{VEI} = \frac{\text{cnt}_k}{\text{total}}. \tag{14}$$

2.1.2. Path Information Feature. In the NDN network, in addition to the feature based on the state of the node, path-based information can be extracted as an aid. To save the path information, a path field needs to be added to the interest packet. This section proposes a hash-based path feature extraction algorithm. The algorithm only uses a few assembly instructions in the operation, almost no reduction in the speed of the original router processing packets. In the memory footprint, the algorithm only needs to add an integer variable in the interest package, and memory also hardly affects network bandwidth.

The algorithm needs to select a random integer as the ID of the router when each NDN router starts, and the path field of the consumer sending interest packet is 0, that is, the content consumer does not participate in the maintenance process of the entire path, if the attacker attempts to change this field to forge the path information, the first hop router can also judge the attacker's attack based on the value being nonzero. The algorithm of the router is as follows:

$$\text{PATH}_{i+1} = \text{PATH}_i \text{ xor } \text{ID}_{i+1}. \tag{15}$$

When the NDN router receives an interest packet, the value of PATH is updated by using equation (15), where PATH_{i+1} represents the PATH value in the interest packet forwarded by the $(i+1)$ th router, ID_{i+1} represents the ID of the $(i+1)$ th router, and xor is the exclusive-or operation. This kind of replacement or filling produces only one assembly code per forwarding time, so it hardly affects the delivery rate of interest packets.

The above PATH value can be approximated to represent the path of the interest packet to a certain terminal, and the definition $\text{Unique}(C)$ indicates the number of different PATH values in the interest packet requesting the content C in the current terminal.$\text{Cnt}(c)$ is defined to represent the number of interest packets that request content C in the current terminal. Obviously, in the case of no cache pollution, there is a positive correlation between the number of interest packets $\text{Cnt}(C)$ and $\text{Unique}(C)$. Therefore,

Table 2: Node characteristics of the model.

Selected features	Description
Cache hit ratio VHit	Cache replacement rate under cache replacement policy
Interest package ratio vector VEI	Proportion of K interest packages with the highest proportion of interest packages
Cache hit rate vector	K cache hit ratios corresponding to interest packages

Unique(C) cannot be directly used as a feature, and Unique(C) needs to be normalized to define the diversification ratio CP(C) of content C as in equation (16). The diversification ratio can reflect the richness of the source of certain content C to some extent, and the diversification ratio can be known according to the definition whose range is between 0 and 1. The smaller the value, the more singular the source of the interest packet is, the more likely it is the attack. The feature has a negative correlation with the cache attack; therefore, the feature can increase the accuracy of the model.

$$\mathrm{CP}(C) = \frac{\mathrm{Unique}(C)}{\mathrm{Cnt}(C)}. \tag{16}$$

It can be known from equation (17) that, to calculate the diversification ratio, CP(C) of content C needs to calculate Cnt(C) and Unique(C), both of which are statistical values. Cnt(C) is the number of interest packets per unit time which requires a numeric variable, and Unique(C) is the number of different kinds of PATH. For an NDN network, considering the network traffic, the interest packets will not be stored, so the hash is required to statistic the values above. Hash the PATH, and, in addition, use the bitmap to reduce the memory usage. Using one bit to represent whether the current PATH has appeared or not; if so, set that bit to 1. Calculate the number of bits with the value of 1 in unit time, which can be approximately considered as the number of different kinds of PATH in network.

3. Experimental Results and Analysis

3.1. Experimental Environment. The experimental environment of this paper is shown in Table 3:

Table 3: Experimental configuration.

Configuration name	Configuration parameter
CPU	Intel I5-8265U
RAM	8G
Operating system	Ubuntu 16.04
Python version	Python 3.6
Python toolkit	Anaconda

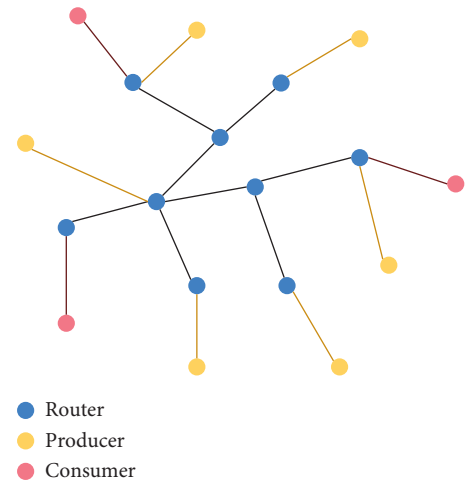


Figure 3: Network topology.

3.2. Experimental Program

3.2.1. ndnSIM Simulation. ndnSIM is an NDN simulation platform developed based on the NS-3 network simulator, which can simulate diversified NDN scenarios [26]. This article changes the source code of the interest package structure in ndnSIM, adds the PATH variable, randomly assigns an ID to each NDN router, and adds related operations to the PATH variable in the routing and forwarding process according to the description above in this chapter. This chapter conducts simulation experiments on known complex topologies. The experimental network topology is shown in Figure 3. In each experiment, the attacker randomly selected the host as the controlled host, and the controlled host sends a great quantity of nonpopular requests.

According to the current researches, most researchers believe that the request in the information-centric network should obey the Zip-f distribution. Therefore, the request in the simulation experiment network should follow the Zip-f distribution, and the normal request distribution's parameter should set $a = 1.2$, and the request rate is 1000/s. In the experiment, the cache policy of the NDN router adopts the LRU policy. The experiment builds the environment through ndnSIM [24], and the statistics of the experimental related data including the number of arrivals of the interest packets are performed by modifying the source code. In order to obtain the training data of GBDT, respectively, simulate the network when there is no attack, and when there is an attack, the attacker sends a large number of nonpopular interest packets to simulate the occurrence of the attack, and statistics when there is an attack and the statistics when there is no attack are recorded and saved separately, and the data are divided into a training set and a test set for multiple experiments. The training set and test set data selection in each experiment are shown in Table 4.

3.2.2. Model Training. This article uses Python's lightGBM library to build the GBDT model. lightGBM is Microsoft's boosting framework, which has faster training efficiency, lower memory usage, and higher accuracy than xgboost [27], and supports parallel learning. In this experiment, the GBDT model is used for training on 10,000 sets of data, and the test

Table 4: Training set and test set data selection.

Dataset category	Positive sample	Negative sample
Training data set	5000	5000
Test data set	1000	1000

is performed on 2,000 sets of data to analyze the accuracy of the model.

When training the model, in order to prevent overfitting of the model, the maximum depth of the decision tree and the maximum number of leaf nodes in the GBDT model should be set, and the regularization parameters should be set. Besides, for the number of iterations, the fast stop strategy is selected, and the training data is divided into two parts: one as the training set and one as the evaluation set (to distinguish it from the test set, here called the evaluation set), used to make a quick stop. The training set and the evaluation set are disjoint sets. In the experiment, their ratio is 4 : 1. The loss function on the evaluation set is calculated in each iteration. When the performance on the evaluation set will not improve anymore (that is, the loss function does not change anymore), stop training and the model loss function uses the Log loss function. When training the GBDT model, some of the parameter settings for using lightGBM are shown in Table 5.

3.2.3. Evaluating Indicator. The validity of the method is measured by the precision, recall rate, accuracy, and F-measure. It defines the following concepts:

Cache pollution is classified as cache pollution: TP (true positive)

Cache uncontaminated is classified as cache uncontaminated: TN (true negative)

Cache uncontaminated is classified as cache pollution: FP (false positive)

Cache pollution is classified as cache uncontaminated: FN (false negative).

Precision (Pre), which is the actual proportion of the sample classified as cache pollution, is calculated as equation (17).

Accuracy (abbreviation: Acc) is the ratio of the correctly classified samples to all samples and measures the overall correctness of the model. The formula is given by equation (18).

The recall rate (recall) indicates how many positive samples are correctly classified, and the formula is calculated as in equation (19).

F-measure is the harmonic mean of the accuracy rate and the recall rate, and the formula is calculated as in equation (20).

Table 5: LightGBM-related parameter settings.

Parameter	Value	Description
Objective	Binary	Two classification problem
Num_leaves	50	Up to 50 leaf nodes per tree
max_depth	8	The maximum depth of the tree is 8
learning_rate	0.01	Learning rate is 0.01
reg_alpha	0.005	$L1$ regular weight is 0.005
reg_lambda	0.0005	$L2$ regular weight 0.0005
Subsample	0.9	Selective feature probability
n_estimators	4000	Maximum number of iterations
early_stopping_rounds	200	Early stopping rounds

$$\text{Pre} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{17}$$

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{18}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{19}$$

$$F1 = \frac{2 \times \text{Pre} \times \text{Recall}}{\text{Pre} + \text{Recall}}. \tag{20}$$

In this paper, the training time of the model is used as the main indicator of model performance evaluation. The relationship between the number of GDBT iterations and time when the experiment counts 10,000 sets of data.

4. Analysis of Results

For the GBDT model proposed in this section, two types of features are used, node status and path information. Since normalization is used, all values are in the range [0, 1], and for the NDN cache pollution attack, the attacker's attack strength will be numerically strong and weak. The characteristics of different attack strengths also change within a certain range. Therefore, the final decision model should be a range model. This property is like the decision tree. GBDT is currently a very good model for improving the decision tree, so the model is adopted. The experiment also proves that the model can achieve good detection results.

Figure 4 shows the relationship between the loss function and the number of iterations when using the parameters described in Section 3 on 10,000 sets of data. It can be seen from the figure that, as the number of iterations increases, the performance of the training set becomes better. However, the performance of the evaluation set is not getting better anymore, and there is a trend of deterioration. If the number of iterations continues to increase, there will be overfitting. In the current model parameters, the loss function of the training set and the evaluation set is better at 736 iterations. At this time, the loss function on the evaluation set is 0.0386, the loss function on the training set is

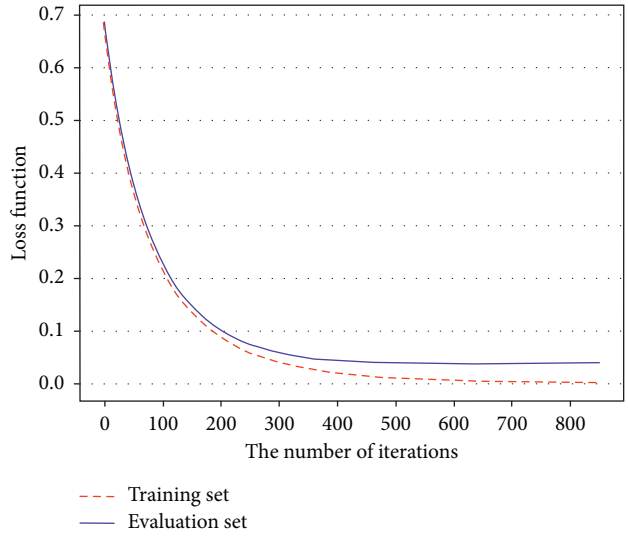


Figure 4: Diagram of the number of iterations and the loss functions.

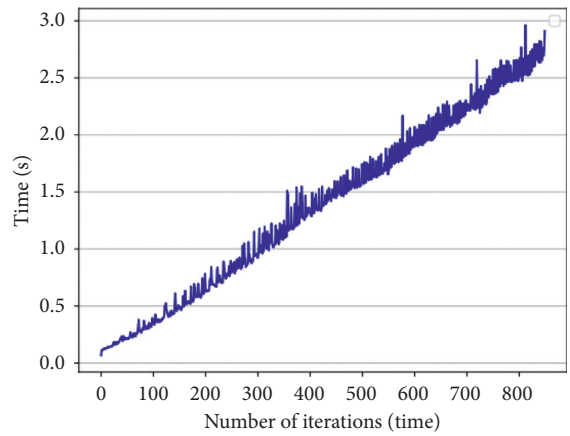


Figure 5: GBDT iteration times and time diagram.

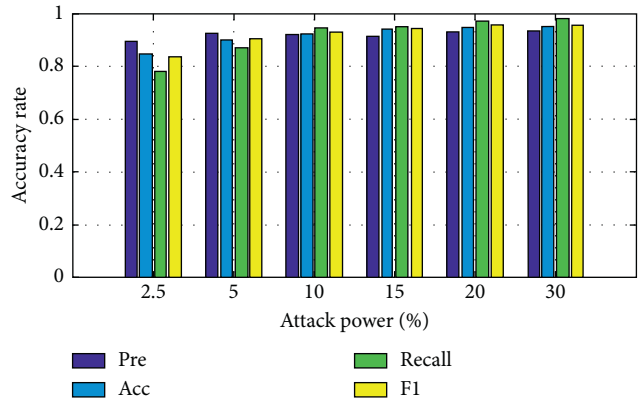


Figure 6: Relationship between attack strength and correctness.

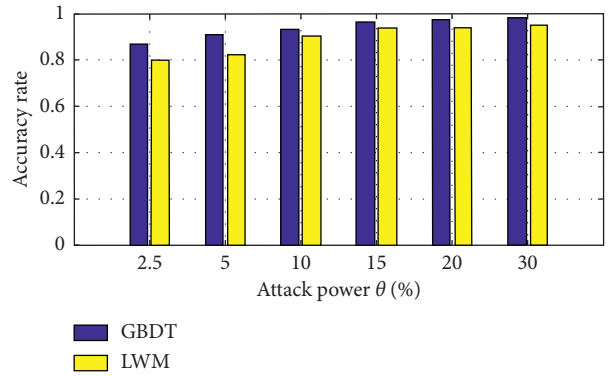


Figure 7: Attack strength and detection accuracy.

0.0029175, and the loss on the test set is 0.015377. Therefore, under the current feature, the model parameters of 736 iterations should be taken.

As can be seen from Figure 5, using the lightGBM for the training of the GBDT model, the training is very fast in the case of 10000 sets of data. When the iteration is about 300 times, the time is still less than 1 second. In the simulation experiment, it is best to iterate. It took only about 2 seconds, which means that the GBDT model training for lightGBM is very fast.

The attack strength θ is defined as the proportion of attack packets in the request packet. The stronger the attack strength is, the greater the impact on the state of the network node is. The accuracy of the model has a certain relationship with the attack strength. Therefore, the simulation is as follows. In the experiment, the relationship between attack intensity and detection accuracy was analyzed.

As shown in Figure 6, when the attack intensity of network cache contamination is lower than 15%, with the continuous increase of the attack intensity, the accuracy rate PRE, accuracy rate Acc, and recall rate of GBDT model detection are all continuously improved. When the attack intensity exceeds 20%, the model can distinguish the attack more clearly, and the above indicators tend to be stable, with the accuracy rate up to 93%, accuracy rate up to 95%, and recall rate up to 97%. This is because with the increase of attack intensity, the cache hit ratio of nodes in the network and the proportion distribution of interest packets will also be affected more and more, which makes it easier for the model to detect attacks.

Figure 7 shows the comparison result of the detection accuracy with the light weight mechanism method proposed in [17]. The traditional LWM method needs to set a threshold, which affects the detection accuracy of the model. The GBDT square model uses the current mainstream machine learning method to learn the judgment standard, so there is no need to set the threshold. It can be seen from the numerical values that the detection accuracy of the GBDT model is higher than that of the LWM method under various attack intensities, and the detection accuracy of the GBDT model can reach more than 85% under the attack intensity of 2.5%, which is better than that of the LWM. The detection accuracy of the method is 5% higher, so it can indicate that the model has a fairly strong cache pollution perception ability.

5. Conclusion

In this paper, a new detection method based on machine learning is proposed for the current problem of cache pollution in the information-centric network. Firstly, the node state and the path information are used as the feature. The node information is obtained through statistics, and the path information is hashed to achieve the diversification rate. In the interest packet delivery process, only one field and one assembly instruction need to be added, so completely it does not affect the forwarding efficiency of normal routes. Then, the two features are combined to use the gradient lifting tree algorithm to build a cache pollution detection model. Compared with the current detection methods that mostly need to set thresholds, the method has higher accuracy and adaptability to different networks. Finally, the model is implemented by ndnSIM and lightGBM, and the advantages of the method in detection accuracy are demonstrated compared with other detection methods.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (grant nos. 61771153, 61831007, and 61971154) and the Fundamental Research Funds of the Central Universities (grant no. 3072020CF0601).

References

[1] M. Amadeo, C. Campolo, J. Quevedo et al., "Information-centric networking for the internet of things: challenges and opportunities," *IEEE Network*, vol. 30, no. 2, pp. 92–100, 2016.

[2] G. Xylomenos, C. N. Ververidis, V. A. Siris et al., "A survey of information-centric networking research," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[3] R. Tourani, S. Misra, T. Mick, and G. Panwar, "Security, privacy, and access control in information-centric networking: a survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 566–600, 2018.

[4] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[5] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pp. 1–12, Rome, Italy, December 2009.

[6] S. Tarkoma, M. Ain, and K. Visala, "The publish/subscribe internet routing paradigm (PSIRP): designing the future internet architecture," in *Future Internet Assembly*, pp. 102–111, IOS press, Amsterdam, The Netherlands, 2009.

[7] T. Koponen, M. Chawla, B. Chun et al., "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 181–192, Kyoto, Japan, August 2007.

[8] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of Information (NetInf) - an information-centric networking architecture," *Computer Communications*, vol. 36, no. 7, pp. 721–735, 2013.

[9] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[10] L. Zhu, X. Tang, M. Shen, X. Du, and M. Guizani, "Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 628–643, 2018.

[11] L. Zhang, A. Afanasyev, J. Burke et al., "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[12] M. Virgilio, G. Marchetto, and R. Sisto, "PIT overload analysis in content centric networks," in *Proceedings of the 3rd ACM SIGCOMM Workshop Information-Centric Network (ICN)*, pp. 67–72, San Jose, CA, USA, August 2013.

[13] T. Nguyen, H.-L. Mai, R. Cogranne et al., "Reliable detection of interest flooding attack in real deployment of named data networking," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 9, pp. 2470–2485, 2019.

[14] H. Park, I. Widjaja, and H. Lee, "Detection of cache pollution attacks using randomness checks," in *Proceedings of the 2012 IEEE International Conference on Communications (ICC)*, pp. 1096–1100, Ottawa, ON, Canada, June 2012.

[15] L. Deng, Y. Gao, Y. Chen, and A. Kuzmanovic, "Pollution attacks and defenses for internet caching systems," *Computer Networks*, vol. 52, no. 5, pp. 935–956, 2008.

[16] M. Li, Y. Sun, H. Lu, S. Maharjan, and Z. Tian, "Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6266–6278, 2020.

[17] L. Yao, Z. Fan, J. Deng, X. Fan, and G. Wu, "Detection and defense of cache pollution attacks using clustering in named data networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 6, pp. 1310–1321, 2020.

[18] G. Mauri, R. Raspadori, M. Gerla, and G. Verticale, "Exploiting information centric networking to build an attacker-controlled content delivery network," in *Proceedings of the 14th IFIP Annual Mediterranean Ad Hoc Networking Workshop*, pp. 1–6, Vilamoura, Portugal, June 2015.

[19] C. Ghali, G. Tsudik, and E. Uzun, "Network-layer trust in named-data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 12–19, 2014.

[20] Q. Li, X. Zhang, Q. Zheng, R. Sandhu, and X. Fu, "LIVE: live: lightweight integrity verification and content access control for named data networking," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 308–320, 2015.

[21] M. Xie, I. Widjaja, and H. Wang, "Enhancing cache robustness for content-centric networking," in *Proceedings of IEEE INFOCOM*, pp. 2426–2434, Orlando, FL, USA, March 2012.

[22] Z. Yi, S. Jia, G. Pu et al., "Collaborative detection mechanism for low-rate cache pollution attack in named data networking," *Journal of Beijing University of Posts & Telecommunications*, vol. 38, pp. 44–48, 2015.

[23] M. Conti, P. Gasti, and M. Teoli, "A lightweight mechanism for detection of cache pollution attacks in Named Data Networking," *Computer Networks*, vol. 57, no. 16, pp. 3178–3191, 2013.

[24] G. Ridgeway, "The state of boosting," *Computing Science and Statistics*, pp. 172–181, Springer, Berlin, Germany, 1999.

[25] L. Breslau, C. Pei, F. Li, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proceedings of the IEEE INFOCOM'99. Conference on Computer Communications. Proceedings Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*, vol. 1, pp. 126–134, New York, NY, USA, March 1999.

[26] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnSIM," *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 19–33, 2017.

[27] T. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, San Francisco, CA, USA, August 2016.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9941464, 9 pages
https://doi.org/10.1155/2021/9941464

Research Article

Explainable Fraud Detection for Few Labeled Time Series Data

Zhiwen Xiao and Jianbin Jiao

School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, BJ10, Beijing, China

Correspondence should be addressed to Jianbin Jiao; jiaojb@ucas.ac.cn

Received 26 March 2021; Revised 21 May 2021; Accepted 6 June 2021; Published 14 June 2021

Academic Editor: Liguo Zhang

Fraud detection technology is an important method to ensure financial security. It is necessary to develop explainable fraud detection methods to express significant causality for participants in the transaction. The main contribution of our work is to propose an explainable classification method in the framework of multiple instance learning (MIL), which incorporates the AP clustering method in the self-training LSTM model to obtain a clear explanation. Based on a real-world dataset and a simulated dataset, we conducted two comparative studies to evaluate the effectiveness of the proposed method. Experimental results show that our proposed method achieves the similar predictive performance as the state-of-art method, while our method can generate clear causal explanations for a few labeled time series data. The significance of the research work is that financial institutions can use this method to efficiently identify fraudulent behaviors and easily give reasons for rejecting transactions so as to reduce fraud losses and management costs.

1. Introduction

Fraud detection is an important technology for identifying abnormal behaviors in the financial field. It aims to detect fraudsters who have no intention to perform and to terminate transactions with potential default risks in advance for avoiding losses. Fraudsters hide deceptive and destructive intentions under disguised compliance behaviors [1] and use flexible and fickle tricks to evade verification from expert experience. Thus, unforeseen frauds pose a serious threat to the normal operation of financial institutions. Therefore, the development of fraud detection technology based on machine learning has gradually become an important consensus in the financial field to reduce financial risks.

Until recently, many technologies based on graph [2], SVM [3], neural networks [4, 5], and even ensemble methods [6] have been developed mature fraud detection mechanism. However, while literatures develop lots of very complex proposals, the financial industry needs comprehensible models to be used in practice, so the empirical usefulness of complex learners is reduced [7]. The reason is that financial institutions must be able to adequately explain the decision it made, especially the reasons for the refusal of the transaction [8]. Providing explainable results is a legal obligation of financial institutions in some countries [9] and an important basis for assisting credit operators in understanding the default factors and making correct decisions. Therefore, financial institutions should consider developing explainable fraud detection methods so as to present all parties to the transaction with significant causal identification results.

However, the known explainable machine learning methods have no impressive performance when applied to fraud identification. The reason is that in most practical financial scenarios, malicious fraud is not a common example. The cost of label collection, especially the collection of default labels, is very expensive. Especially for financial scenarios with a long transaction cycle, institutions have to wait until the end of the contract to fully affirm their willingness to perform in the transaction, so as to obtain a positive posterior label [10]. It means that in a large amount of data derived from historical transaction records, there are only a relatively small number of "good" samples and a smaller number of "bad" samples and most of the remaining samples are unlabeled. Therefore, it is difficult to develop fraud detection technology based on supervised learning on historical transaction data of known labels. In addition,

unsupervised technology does not require knowledge of labels. However, most unsupervised methods are based on the assumption that fraud performance is an outlier in the distribution of transaction behavior data [11]. This assumption weakens the ability to recognize deliberate concealment or disguise. In summary, it is worth exploring to construct an explainable fraud detection classifier by integrating the complementary methods of these two technologies.

Our main contribution is to propose an explainable classification method by improving the multiple instance learning (MIL) framework so as to realize fraud detection for time series data with few labeled. Different from traditional methods, the one-to-one correspondence between samples and labels is not sensitive in MIL. Under the improved MIL framework, fraud detection on tracklets of time series data can obtain acceptable explanations for the prediction of each sample.

The remainder of this paper is organized as follows. Section 2 reviews the principles of MIL. We introduce the details of the method to enhance the explainability of MIL in Section 3. In Section 4, we verify the performance of the proposed method and compare it with some existing explainable technologies. Finally, our conclusions are provided in Section 5.

2. Related Work

MIL is a relatively novel weakly supervised machine learning method. It can achieve considerable performance when training datasets with poor label quality [12]. The MIL method arranges the training set in several labeled groups, which are called instance bags, and builds a classifier for the labels of bags. MIL treats a single instance in the training set as a subvector of the feature vector set in bags and only supervises the entire multibag. The training dataset of MIL can be expressed as follows:

$$X = \{(B_1, Y_1), (B_2, Y_2), \ldots, (B_n, Y_n)\}, \tag{1}$$

where $B_n \subseteq B$ represents the nth bag in the dataset and $Y_n \in \{-1, 1\}$ represents the label of the bag B_n in the binary classification problem.

Since the MIL method was proposed, research studies on this theoretical framework have produced many technologies. In the process of performing the classification task, the MIL method can be divided into two categories according to the position of extracting the feature information from the bag-space learning paradigm and the instance-space learning paradigm.

2.1. The Bag-Space Learning Paradigm. The bag-space learning paradigm takes each bag as an independent individual to extract information and assign class labels. Based on the global information in the bag level space, the bag-space learning paradigm tries to find a hyperplane that can separate the bags in the nonvector space so as to achieve an effective classification. For this reason, many research studies have focused on how to measure the distance or similarity between arbitrary bag-spaces. The Hausdorff distance [13], which is the Euclidean distance between the closest instance in two bags, was introduced to measure the distance between bags. Subsequent classification research at the bag level derived the embedded-space learning paradigm, which maps the bag-space to a single feature vector. The feature vector tried to express the whole information about a bag, and each feature vector has an associated label. In this paradigm, the original bag-space is mapped to an embedding space vector, and the classifier is trained in this new space. It converts the original problem into a standard supervised learning problem effectively and then applies any standard classifier for training. In the process of mapping bag-space to vector, the dimensionality of the embedding space is much higher than the number of training bags. Therefore, the study of embedding space learning has focused on the feature selection [14, 15].

2.2. The Instance-Space Learning Paradigm. The instance-space learning paradigm is based on the classifier in the instance space. After the instances have the classification result, the label of the package is determined according to the concept (label of instance). This paradigm infers from the instance level label to the bag level label, and there is an assumption about the relationship between the bag label and the instance label in the training set. In the standard MIL assumption, each instance has a hidden label. If and only if the bag contains at least one positive instance, the bag is marked as a positive bag. If and only if all the negative instances in the bag are negative, the bag is marked as a negative bag. The multi-instance bag classifier $F(B_i)$ is expressed by the following equation:

$$F(B_i) = \begin{cases} +1, & \exists x_n^i \in B_i : f(x) = +1, \\ -1, & \text{otherwise}, \end{cases} \tag{2}$$

where x_n^i is the nth instance in the bag B_i and $f(x)$ represents the discriminant function for inferring the label of the instance in the feature space. The generalization of the standard hypothesis leads to the collective assumption [16]. The label of a bag is determined by the multiclass label (concept) in the instance level. The expression is shown by the following equation:

$$F(B_i) = \begin{cases} +1, & \forall c \in \Pi^+ : \sigma_c \leq \sum_{x \in B_i} f_c(x) = 1, \\ -1, & \text{otherwise}, \end{cases} \tag{3}$$

where c is a concept, $\Pi^+ \in \Pi$ is a collection of positive concepts, σ is a predefined threshold, and $f_c(x)$ is the discriminant function under the concept c. Based on the above assumption, the classic mi-SVM method [17] trained a SVM classifier to update the label after the instance label has been initialized. This step is performed until the label no longer changes. The trained classifier is used to predict the label of instances. The improved method [18] calculated the probability of instance selection based on the training data gathered in a random subspace and used these probabilities to create a classifier pool for training subinstances. This

method does not need to make any prior assumptions about the data structure and the proportion of instances in the bag.

When predicting time series data based on the MIL method, the similarity loss across bags [19] is introduced to model the sequential constraints between the news published on different days. This approach minimizes the total loss to obtain the probability of each news being a precursor. An improved MIL method [20] based on radial basis function (RBF) extracts features from transaction data to predict the likelihood of default based on behavior. An extend MIL method [21] was proposed to evaluate credit scores by transactional data and static individual information. This method considers the dynamic transactional data and cost-sensitive problem simultaneously. Essentially, the bag level label in MIL is pushed by the subinstance level label, and then the causal relationship between a bag label and subinstance label can be expressed by backtracking from the bag-space to the instance space. This shows that the instance-space learning paradigm is easier to enhance the explainability of the MIL algorithm. However, no relevant research has attempted to explore the prediction of time series data by the MIL methods from the perspective of enhancing model explainability. The main reason is that insufficient instance labels cannot be inferred using effective supervised learning methods, and it is not easy to train a classifier in the instance space. In summary, in the instance-space learning paradigm, how to obtain an instance label is the key to an explainable MIL classifier.

3. MIL Classification Method with Enhanced Explainability

In this section, the explainable MIL framework adopts the instance-space learning paradigm and a self-training semisupervised learning method. The calculation framework is shown in Figure 1:

Aiming at the learning of few labels at the instance space, the proposed method regards each input sample as a bag and splits the original behavior trajectory of each sample into tracklets at a specific time interval as a feature space in instance level. Each tracklet is an instance in the bag. We use the affinity propagation (AP) algorithm to improve the self-training method based on the long short-term memory (LSTM) model. It iteratively trains the classifier by learning the global information in the instance space and marks unlabeled instances.

In the initialization of the instance label Y^{ins}, the proposed method takes the label of the original training set as the bag label Y^{bag} and assigns it to all the instances in this bag. Among them, the instances in the unlabeled package are initialized as negative instances (no fraud). Clustering methods fit well for obtaining the hidden structure information in feature space [22]. Considering unlabeled tracklets, we cluster behavior tracklets on a low-dimensional manifold space. The clustering results are used to determine the behavior clusters of each tracklet; then, the existing labeled tracklets are used in the cluster to define the label of all tracklets in this cluster. The clustering diagram of the AP algorithm in the instance space is shown in Figure 2.

The clustering result makes the tracklet X^{ins} gather in a cluster $C_z \in C$ with the same behavior. Then, each instance represented by the tracklet can be labeled by clustering. The instance of cluster center is considered to be a behavior prototype $P_z \in P$.

When using the AP algorithm to perform unsupervised clustering of instances, the similarity between input data points is used as a clustering measure. The AP algorithm records the similarity in a matrix S, and the expression of elements in the matrix is shown as follows:

$$s(i,k) = -x_i - x_k^2. \tag{4}$$

The element $s(i,k)$ can be regard as the distance from the kth data point to the ith data point which can be a cluster center point (prototype). The AP algorithm does not preset the number of clusters and cluster centers in the initial stage but treats all data as candidate cluster centers and selects high-quality cluster centers as prototype instances through information exchange. The selection process of the cluster center point is realized by the matrix R (which represents responsibility) and the matrix A (which represents availability) where the element $r(i,k)$ in matrix R is defined as the degree that how the data point k can be used as the cluster center of the data point i. The element $a(i,k)$ in matrix A is defined as the degree that how the data point i selects the data point k as its cluster center suitability. The iterative process is as follows:

$$r_{t+1}(i,k) \longleftarrow \begin{cases} s(i,k) - \max\limits_{j \neq k}\{r_t(i,j) + a_t(i,j)\}, & i \neq k, \\ s(i,k) - \max\limits_{j \neq k}\{S(i,j)\}, & i = k, \end{cases} \tag{5}$$

$$a_{t+1}(i,k) \longleftarrow \begin{cases} \min\left\{0, r_{t+1}(k,k) + \sum\limits_{j \neq i,k} \max\{0, r_{t+1}(j,k)\}\right\}, & i \neq k, \\ \sum\limits_{j \neq i,k} \max\{0, r_{t+1}(j,k)\}, & i = k. \end{cases} \tag{6}$$

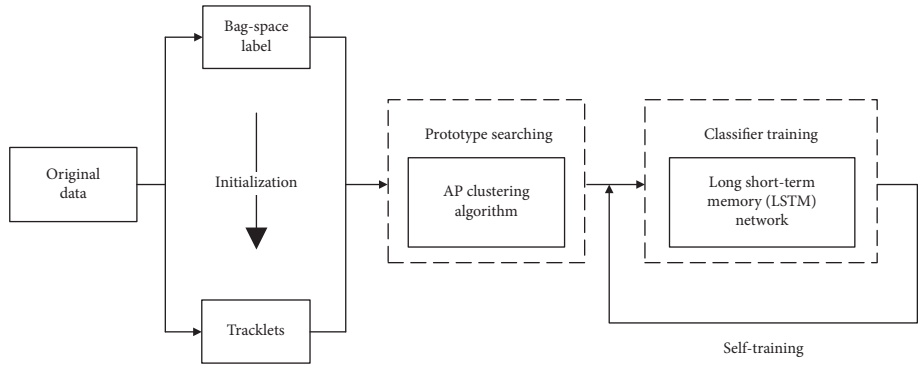


Figure 1: The computational framework of the MIL method with enhanced explainability.

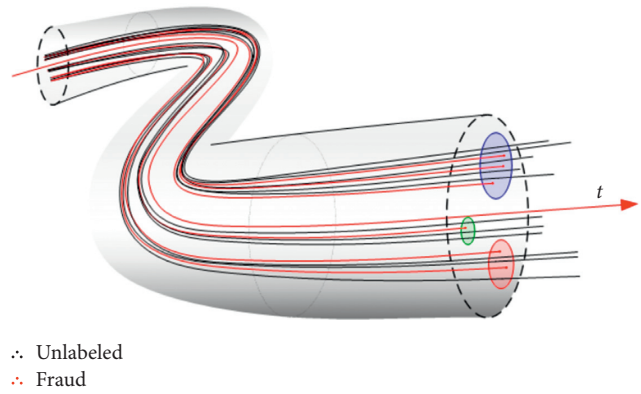


Figure 2: Low-dimensional manifold clustering of behavior tracklets in the instance space.

In the process of iterate operators, the damping factor $\lambda \in (0, 1)$ is introduced to perform a weighted summation of the values before the operator iteration. It preserves the effective information generated during the last iteration and avoids the numerical oscillations during the iteration. The weighting calculation is shown in the following equations:

$$r_{t+1}(i,k) = \lambda \times r_t(i,k) + (1-\lambda) \times r_{t+1}(i,k), \quad (7)$$

$$a_{t+1}(i,k) = \lambda \times a_t(i,k) + (1-\lambda) \times a_{t+1}(i,k). \quad (8)$$

After the iteration, for any data point i, the maximum in $r_t(i,k) + a_t(i,k)$ of the data point k is selected as a cluster center point. Therefore, the element $S(k,k)$ on the main diagonal of the similarity matrix is bias parameters, which should be set to a larger value.

Next, we train the LSTM classifier iteratively. In each iteration, we train the LSTM classifier M_c and predict instance to update Y^{pre} after training. In the next iteration, each instance label-set Y_i^{pre} from a negative bag will be corrected to a negative label $y_i^{\text{bag}-}$ of its bag. Then, we use the cluster prototype label to correct the label of the instances in this cluster. Finally, the classifier M_c is retrained and updated in the training set with the updated labels. The self-training process is stopped when the prototype label-set Y^{pre} changes no longer significantly.

The pseudocode of the self-training method is shown in Algorithm 1:

We improve the hypothesis based on the inference method of fraud in the actual field. The hypothesis is that when at least one kind fraud is detected in the transaction, the sample is a positive instance of fraud. The classifier $F(B_i)$ is shown as the following equation:

$$F(B_i) = \begin{cases} +1, & \exists x_n^i \in B_i : f(x) \neq -1, \\ -1, & \text{otherwise.} \end{cases} \quad (9)$$

It can be observed from the expression that the proposed learning framework obtains the bag label by predicting the label of the tracklets. So, the instance label can be the reason for the bag label. The causality can be explained as follows: when the label of the sample is negative, it means that no abnormal behavior is detected; when the label of the sample is positive, there has been a fraudulent behavior belonging to a certain prototype in the sample at least for a period of time. In summary, when classifying time series datasets

Input: Instance-set X^{ins} and bag label-set $Y^{\text{bag}} = \left\{y_1^{\text{bag}}, y_2^{\text{bag}}, \ldots, y_n^{\text{bag}}\right\}$
Output: Instance label-set $Y^{\text{ins}} = \{y_1^{\text{ins}}, y_2^{\text{ins}}, \ldots, y_n^{\text{ins}}\}$
X^{ins} clustering by the AP algorithm;
Get cluster $C = \{C_1, C_2, \ldots, C_z\}$ and instance prototype $P = \{P_1, P_2, \ldots, P_z\}$;
Initialize instance label $Y_i^{\text{ins}} \longleftarrow y_i^{\text{bag}}$ $(i \le n)$;
repeat
$Y_i^{\text{pre}} \longleftarrow y_i^{\text{bag}-}$ p.s. negative bag label correction
$Y_{C_k}^{\text{pre}} \longleftarrow y_{P_k}^{\text{pre}}$ p.s. prototype label correction
LSTM classifier M_c training in dataset (X, Y^{pre})
Predict X^{ins} by M_c, update Y^{pre}
until Y^{pre} not change

Algorithm 1: The self-training algorithm of the LSTM classification model.

represented by transaction behavior data, the expression of causality is the description of features of tracklets in the time-space. In the proposed ML framework, the MIL method can propose an explanation for bag label by predicting instances label, thereby improving the explainability of classification results for few labeled time series data.

4. Experimental Results and Analysis

In this section, we evaluate the performance of our proposed explainable method. We try to verify and answer two main questions through experiments. The first question is whether this method can provide better performance for real-world fraud detection tasks. The second question is whether this method can maintain a considerable performance in the few labeled training set. For this purpose, we have selected four classic methods as benchmarks for comparison with the proposed methods, namely: SVM, random forest (RF), AP clustering algorithm, and HOBA [5] method where the SVM technique is the most widely financial fraud detection technique used in data mining [23] and the RF model has good performance in many classification problems [24]. The AP clustering algorithm was selected to verify the effectiveness of the self-training process in our proposed method. The recently developed HOBA (homogeneity-oriented behavior analysis) has achieved outstanding experimental performance comparing with many related studies.

In order to compare the performance of these classifiers intuitively, we have selected representative metrics for commonly evaluating classifiers. Accuracy (Acc) is a standard performance indicator used to compare classifiers, $F1$-score ($F1$) is the harmonic mean of Precision and Recall, and AUC represents the area under the ROC curve. Because AUC does not include category distribution or misclassification costs, it is widely used to evaluate models trained on unbalanced datasets. In the classification problem, the calculation of AUC refers to the existing method, and the calculation is shown in the following equation:

$$\text{AUC} = \frac{1}{2}(1 + \text{TPR} - \text{FPR}), \tag{10}$$

where TPR is the true positive rate and FPR is the false positive rate. These three metrics can reflect the overall performance of the model. In addition, Recall and Precision are both important evaluation metrics in fraud detection tasks. Recall can reflect the ability to identify fraud risks, while Precision can reflect the discrimination cost of the classifier. The experiment compares the metrics of the proposed method with that of other benchmark classifiers to fully observe the performance of the proposed method.

We used different datasets to verify the two problems (details in this section below), but two datasets were preprocessed in the same way. We excluded duplicates, outliers, and accounts with no transactions in datasets. We sum the transaction data by date according to the timestamp and generate the two-dimensional feature vector of the Month × Date as the input of the proposed method. Furthermore, when training these benchmark classifiers, we have to reduce the dimensionality of two datasets to match the input requirements of benchmark classifiers. The detailed results of the two experiments will be introduced separately.

4.1. Performance Analysis in Fraud Detection. For the first question, we used a private credit card transaction details provided by an anonymous financial institution. Some fraudulent transactions were marked based on real investigations during the performance period, and the sample labels are incomplete. The dataset used in the experiment included a total of more than 5 million transaction records of 8057 accounts in 573 days, of which 1228 accounts (15.2%) show clear fraud during the performance period and 537 accounts have good performance labels and remaining accounts are unlabeled. We selected 100 positive samples and 100 negative samples with clear labels as the validation set and the remaining data as the training set.

After ten independent runs in different data partitions, the experimental results of each method are statistically analysed by the average values. The proposed method is compared with other comparison methods in identifying fraud categories (positive instances). The results are shown in Table 1.

In order to compare the performance of each classifier more intuitively, we show the fitted ROC curve in Figure 3.

We can observe that the improved MIL method based on self-training is close to the performance of the compared HOBA method. The performance of the RF model on this

Table 1: The experimental result on the real-world dataset.

	$F1$	Acc ± std (%)	Recall (%)	Precision (%)	AUC
AP	0.5911	58.58 ± 1.24	61.25	58.25	0.5852
SVM	0.5698	61.24 ± 3.96	52.15	64.58	0.6788
RF	0.6596	67.35 ± 0.56	63.28	69.37	0.6122
HOBA	0.8377	84.52 ± 1.34	80.25	87.91	0.8424
Ours	0.8317	82.53 ± 1.56	84.13	82.39	0.8360

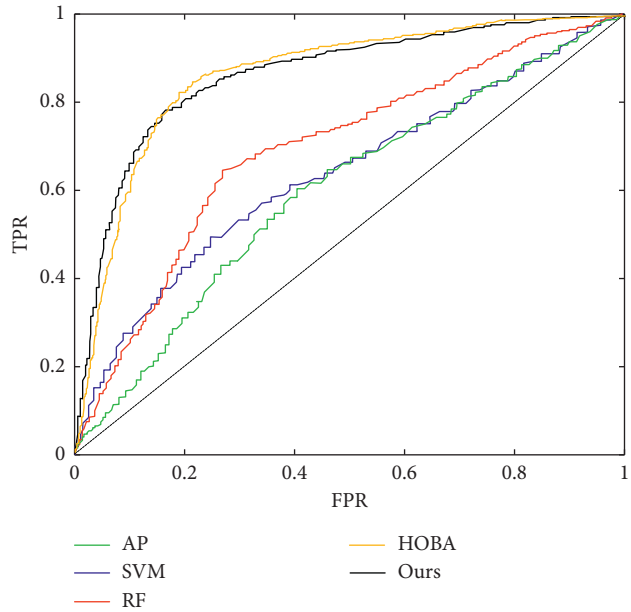


Figure 3: ROC curve of each classifier on the real-world dataset.

dataset is still better than that of the other two traditional methods, which is consistent with most previous research results. $F1$-score, accuracy, and AUC value can reflect that the overall performance of the two methods which are significantly ahead of other classifiers. Among them, the proposed method has the highest Recall rate; it means the proposed method is more conducive to the financial business that cannot accept false rejections. The comparison results show that it is difficult to classify transaction data composed of time series data by using traditional methods. Our proposed method achieves considerable performance almost the same as the state-of-the-art method when dealing with such tasks, and it is worth mentioning that our method is explainable. In summary, the effectiveness of the proposed explainable method is verified in predicting actual time series data.

4.2. Performance Analysis in Few Labeled Dataset. For the second question, we compared the influence of the number of labels in the same dataset on our proposed method. Due to the inherent privacy nature of financial transactions, no transaction dataset is legally published. This prevents us from collecting sufficient labeled transaction data. For this reason, our comparative experiment is verified on the dataset generated by the PaySim simulator. The PaySim simulator based on the agent-based simulation technology framework, combined with the application of mathematical statistics [25], proved that the simulation data can be used as the original dataset for research. The generated dataset contains a total of 1,852,392 transaction records from nearly 1,000 accounts for more than 700 days, of which 9,651 fraudulent transaction data (0.5%) were randomly mixed. The mixing of fraudulent data has resulted in 25% of the samples being fraudulent. The dataset divides the last 6 months of transaction details into verification set. The training set contains 1,296,674 transaction records from 870 accounts, while the test set contains 555,718 transaction records from 218 accounts.

We randomly hide the labels of 50% of the accounts to construct a compared dataset. Considering that each benchmark classifier is based on supervised learning, we only input labeled data for the benchmark learner and input all data for the proposed method. After ten independent runs in different data partitions, the experimental results of each method are statistically analysed by the average values. The performance of each classifier on the PaySim simulation dataset is shown in Table 2.

Table 2: The experimental results on the PaySim simulation dataset.

	$F1$	Acc ± std (%)	Recall (%)	Precision (%)	AUC
AP	0.5481	80.22 ± 2.87	59.47	50.82	0.7006
SVM	0.5911	75.68 ± 2.39	48.91	69.17	0.7343
RF	0.6837	82.74 ± 1.39	59.60	80.18	0.8171
HOBA	0.6439	83.95 ± 2.12	63.23	66.67	0.7658
Ours	0.7789	83.63 ± 1.08	59.04	98.72	0.8910

Table 3: The experimental results on the PaySim simulation dataset with missing labels.

	$F1$	Acc ± std (%)	Recall (%)	Precision (%)	AUC
AP	0.4233	60.49 ± 3.74	32.28	61.46	0.6083
SVM	0.5911	51.57 ± 2.42	25.32	54.13	0.5275
RF	0.6837	58.96 ± 2.47	31.22	61.46	0.5982
HOBA	0.3543	53.46 ± 1.74	54.12	27.18	0.5346
Ours	0.7789	84.58 ± 2.27	61.92	86.69	0.8511

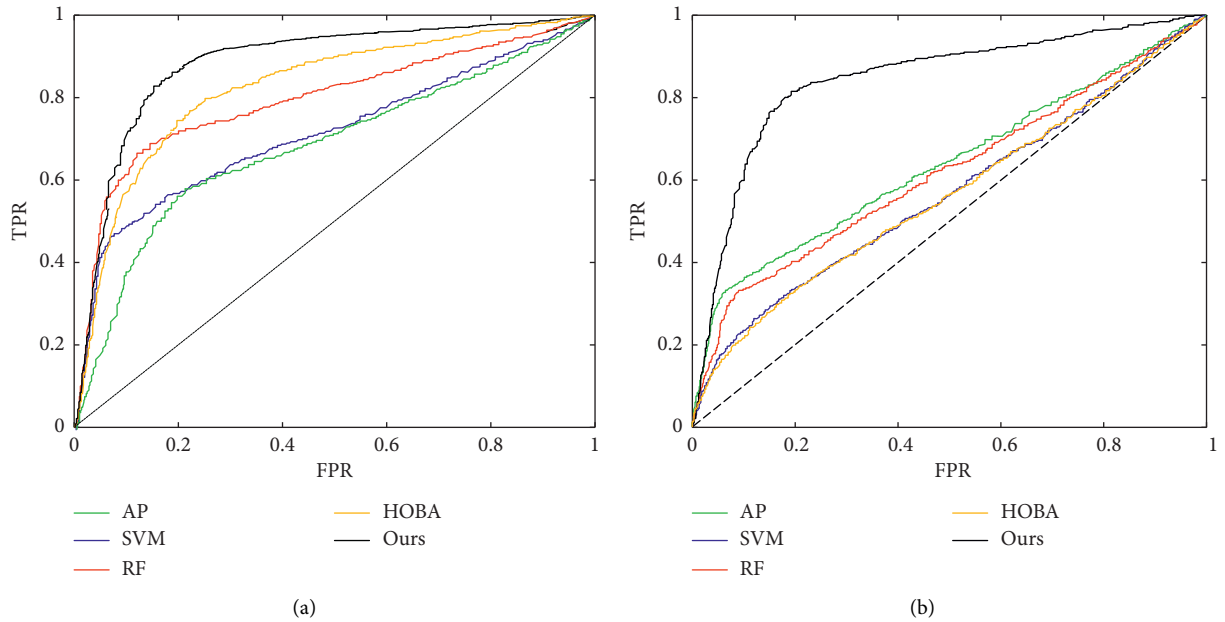


Figure 4: (a) ROC curve of each classifier on the PaySim simulation dataset; (b) ROC curve of each classifier on the PaySim simulation dataset with missing labels.

The performance of each classifier on the PaySim simulation dataset with 50% missing labels is shown in Table 3.

The fitted ROC curves of each classifier on the two comparison datasets are shown in Figure 4.

We observe that our proposed method provides better experimental results on the PaySim simulation dataset. The performance of the HOBA method is much better than that of the other three benchmark classifiers. The proposed method has the highest Precision rate in experiments. It proves that the proposed model can detect more than half of the suspected fraud under the expected low FPR rate. After some labels were hidden, the performance of all benchmark classifiers dropped significantly and the performance of the HOBA method decreased the most. However, the proposed method still maintains significant performance on the dataset with missing labels, which shows that our improved self-training model effectively learns the hidden fraud features in the dataset through a semisupervised method. From these experimental results, it can be observed that the LSTM model of self-training in the MIL framework has stronger fraud detection capabilities for few labeled data in the real financial field.

Table 4: The explainable results of the proposed classifier.

Sample ID	Predicted label	Explanation
****6293	Compliance	No abnormal behavior detected
****8428	Compliance	No abnormal behavior detected
****7570	Fraud	Type III prototype in August
****2085	Fraud	Type VI prototype in May and type II prototype in June

4.3. Explanation Analysis. The method we propose can give intuitive and concise reasons for the classification prediction results of every testing samples. The explainable predicted results of several examples are shown in Table 4.

Among them, when the label is Compliance, there is only one reason for no abnormal behavior detected. However, there are many reasons for Fraud label; this is a combination of multiple fraud categories. From the MIL framework, we observed that the category label of the sample is determined by the multi-instance bag label, and the reason is the concept in bag, for example, sample with ID ****7570; its bag label is Fraud; the reason is expressed as "August feature in the bag is marked as type III." It means that a sample with with ID ****7570 transaction behavior is suspected to be a fraudulent prototype of Type III in August. Therefore, our proposed method provides an explainable prediction method for the real-world time series data through the self-training LSTM prediction model with the AP clustering algorithm in the MIL framework. In addition, with the in-depth application of the proposed method, fraud prototypes can introduce descriptions based on expert experience to achieve a more vivid explanation of the predicted results.

5. Conclusions

In this paper, we proposed a fraud detection method with enhanced explainability in the MIL framework, which incorporates the AP clustering method in the self-training LSTM prediction model. Compared with previous work, we focus on the actual problems of real financial data and obtain a classifier with high predictive performance and clear causal explanation on a few labeled dataset.

The empirical research is based on two datasets, compared with three benchmark classifiers and variated the proposed method from two aspects. First, the real dataset from an anonymous organization is used to evaluate the overall performance of the proposed method. Compared with other classifiers, the proposed method is more effective in predicting actual transaction data. Then, the data generated by the PaySim simulator are used to verify the performance changes in the case of hiding labels. When 50% of account labels are artificially hidden, the proposed method still maintains good predictive performance even when the benchmark classifiers generally drop in performance. It verifies that the proposed method can effectively learn and distinguish the fraud features hidden in the dataset. The empirical analysis results provide trustable evidence, which proves in two steps that our proposed method can complete the classification task with significant performance advantages.

As far as we know, among many fraud detection methods for transaction data, this research is one of the few classification techniques that can obtain a clear casual explanation. The significance of our work is that financial institutions can efficiently identify fraudulent behaviors and easily give reasons for rejecting of transactions so as to reduce the fraud losses and management costs. However, our work still has limitations in the prediction problem for large-scale datasets. The complexity of the AP algorithm leads to higher requirements for computing resources. Therefore, in future work, we hope to explore possible combinations of more advanced clustering algorithms and deep learning to develop more efficient fraud detection methods.

Data Availability

The desensitized sensitive data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Y. Wu, Y. Xu, and J. Li, "Feature construction for fraudulent credit card cash-out detection," *Decision Support Systems*, vol. 127, Article ID 113155, 2019.

[2] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud detection: a systematic literature review of graph-based anomaly detection approaches," *Decision Support Systems*, vol. 133, Article ID 113303, 2020.

[3] N. Rtayli and N. Enneya, "Enhanced credit card fraud detection based on SVM-recursive feature elimination and hyper-parameters optimization," *Journal of Information Security and Applications*, vol. 55, Article ID 102596, 2020.

[4] A. Rb and S. K. Kr, "Credit card fraud detection using artificial neural network," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 35–41, 2021.

[5] X. Zhang, Y. Han, W. Xu, and Q. Wang, "HOBA: a novel feature engineering methodology for credit card fraud detection with a deep learning architecture," *Information Sciences*, vol. 557, pp. 302–316, 2021.

[6] S. Bagga, A. Goyal, N. Gupta, and A. Goyal, "Credit card fraud detection using pipeling and ensemble learning," *Procedia Computer Science*, vol. 173, pp. 104–112, 2020.

[7] R. Florez-Lopez and J. M. Ramon-Jeronimo, "Enhancing accuracy and interpretability of ensemble strategies in credit risk assessment. A correlated-adjusted decision forest proposal," *Expert Systems with Applications*, vol. 42, no. 13, pp. 5737–5753, 2015.

[8] J. N. Crook, D. B. Edelman, and L. C. Thomas, "Recent developments in consumer credit risk assessment," *European Journal of Operational Research*, vol. 183, no. 3, pp. 1447–1465, 2007.

[9] M. B. Gorzałczany and F. Rudziński, "A multi-objective genetic optimization for fast, fuzzy rule-based credit classification with balanced accuracy and interpretability," *Applied Soft Computing*, vol. 40, pp. 206–220, 2016.

[10] P. Jrana and J. Baria, "A survey on fraud detection techniques in ecommerce," *International Journal of Computer Applications*, vol. 113, no. 14, pp. 5–7, 2015.

[11] F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Information Sciences*, vol. 557, pp. 317–331, 2019.

[12] M.-A. Carbonneau, V. Cheplygina, E. Grange, and G. Gagnon, "Multiple instance learning: a survey of problem characteristics and applications," *Pattern Recognition*, vol. 77, pp. 329–353, 2017.

[13] Z. Zhouyu Fu, A. Robles-Kelly, and J. Jun Zhou, "MILIS: multiple instance learning with instance selection," *Ieee*

Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 958–977, 2011.

[14] L. Yuan, J. Liu, and X. Tang, "Combining example selection with instance selection to speed up multiple-instance learning," *Neurocomputing*, vol. 129, pp. 504–515, 2014.

[15] L. Yuan, X. Wen, H. Xu, and L. Zhao, "Multiple- instance learning with empirical estimation guided instance selection," in *Proceedings of 2018 24th International Conference on Pattern Recognition*, pp. 770–775, Beijing, China, August 2018.

[16] J. Foulds and E. Frank, "A review of multi-instance learning assumptions," *The Knowledge Engineering Review*, vol. 25, no. 1, pp. 1–25, 2010.

[17] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proceedings of the 15th International Conference on Neural Information Processing Systems*, pp. 577–584, MIT press, Cambridge, MA, USA, January 2002.

[18] M.-A. Carbonneau, E. Granger, A. J. Raymond, and G. Gagnon, "Robust multiple-instance learning ensembles using random subspace instance selection," *Pattern Recognition*, vol. 58, pp. 83–99, 2016.

[19] Y. Ning, S. Muthiah, H. Rangwala, and N. Ramakrishnan, "Modeling precursors for event forecasting via nested multi-instance learning," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1095–1104, San Francisco, CA, USA, 2016.

[20] T. Zhang, W. Zhang, W. Xu, and H. Hao, "Multiple instance learning for credit risk assessment with transaction data," *Knowledge-Based Systems*, vol. 161, pp. 65–77, 2018.

[21] W. Zhang, "Cost-sensitive multiple-instance learning method with dynamic transactional data for personal credit scoring," *Expert Systems with Applications*, vol. 157, Article ID 113489, 2020.

[22] C. He, J. Shao, J. Zhang, and X. Zhou, "Clustering-based multiple instance learning with multi-view feature," *Expert Systems with Applications*, vol. 162, Article ID 113027, 2020.

[23] K. G. Al-Hashedi and P. Magalingam, "Financial fraud detection applying data mining techniques: a comprehensive review from 2009 to 2019," *Computer Science Review*, vol. 40, Article ID 100402, 2021.

[24] H. D. Nayak, F. Deekshita, L. Anvitha, A. Shetty, D. J. D'Souza, and M. P. Abraham, "Fraud detection in online transactions using machine learning approaches-a review," *Advances in Intelligent Systems and Computing*, pp. 589–599, 2021.

[25] E. Lopez-Rojas, A. Elmir, and S. Axelsson, "Paysim: a financial mobile money simulator for fraud detection," in *Proceedings of the Annual Simulation Symposium*, pp. 249–255, Larnaca, Cyprus, September 2016.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6663028, 13 pages
https://doi.org/10.1155/2021/6663028

Research Article

Project Gradient Descent Adversarial Attack against Multisource Remote Sensing Image Scene Classification

Yan Jiang, Guisheng Yin, Ye Yuan, and Qingan Da

College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Guisheng Yin; yinguisheng@hrbeu.edu.cn

Received 27 December 2020; Revised 7 February 2021; Accepted 3 June 2021; Published 12 June 2021

Academic Editor: Mohamed Amine Ferrag

Deep learning technology (a deeper and optimized network structure) and remote sensing imaging (i.e., the more multisource and the more multicategory remote sensing data) have developed rapidly. Although the deep convolutional neural network (CNN) has achieved state-of-the-art performance on remote sensing image (RSI) scene classification, the existence of adversarial attacks poses a potential security threat to the RSI scene classification task based on CNN. The corresponding adversarial samples can be generated by adding a small perturbation to the original images. Feeding the CNN-based classifier with the adversarial samples leads to the classifier misclassify with high confidence. To achieve a higher attack success rate against scene classification based on CNN, we introduce the projected gradient descent method to generate adversarial remote sensing images. Then, we select several mainstream CNN-based classifiers as the attacked models to demonstrate the effectiveness of our method. The experimental results show that our proposed method can dramatically reduce the classification accuracy under untargeted and targeted attacks. Furthermore, we also evaluate the quality of the generated adversarial images by visual and quantitative comparisons. The results show that our method can generate the imperceptible adversarial samples and has a stronger attack ability for the RSI scene classification.

1. Introduction

With the high-speed development of remote sensing technology, we can obtain more multisource and multicategory remote sensing images (RSIs). Those images are more efficiently employed to observe ground objects. RSI encompasses more the abundant spectral features of the ground objects, i.e., the texture and geometric features. Therefore, employing RSI to achieve scene classification has also gathered considerable attention. The purpose of scene classification is to predict the label for a given RSI, e.g., airport, forest, and river. The critical process of RSI scene classification is to extract image features. The existing researches can be mainly divided into two strategies according to the features used: (1) the low-level features and the middle-level global features obtained by the manual extraction and (2) the high-level features extracted automatically by the convolutional neural network (CNN).

In early RSI scene classification research, scene classification mainly relies on handcrafted low-level and middle-level features [1–5], e.g., color, texture, spatial structure, and scale-invariant feature transform [6]. The middle-level features' extracting process can be regarded as the high-order statistical of the low-level local features [7–9]. Compared with the low-level features, the middle-level features (e.g., bag-of-visual-words model [10]) have expressive content for scene images. However, due to the lack of flexibility and adaptability, using the middle-level features cannot easily distinguish the complex scenes and gain the ideal classification accuracy. Traditional methods can express RSI features to a certain extent [11]. Nevertheless, the above features are manually extracted, and end-to-end scene classification cannot be achieved. The final classification accuracy is mainly subject to the human experience.

To ameliorate the limitations of handcrafted features, more researchers adopted the automatic extracting of RSI features. In recent years, deep learning has achieved

impressive performance in image processing tasks, e.g., image generation, object detection, and image classification. Moreover, the CNN-based classier has also been employed for RSI scene classification. With an end-to-end framework, CNN performs multilayer nonlinear transformations to extract the high-level image features automatically. These features focus on the overall semantics of RSI rather than the pixels of a local region. Hence, the high-level features obtained by CNN can be more effective for RSI scene classification. For instance, Wang et al. [12] proposed a method based on Residual Network (ResNet) to achieve RSI classification. It breaks the limitation of lacking training samples. Zhang et al. [13] proposed an effective architecture based on CNN and Capsule Network (CapsNet), named CNN-CapsNet. RSI scene classification has a common problem, i.e., the overfitting caused by the deeper network (too many parameters). However, the shallower network has not enough extraction ability of semantic information. To this end, Zhang et al. [14] proposed a simple and efficient network based on the dense convolutional network (DenseNet). DenseNet can improve network performance without increasing the number of network parameters.

Although the above CNN-based methods have an outstanding performance for RSI scene classification, they also face various security risks [15], e.g., data poisoning attacks [16] and adversarial sample attacks. To defend against these attacks, researchers also proposed corresponding malicious detection methods [17–19]. Among these attacks, the classifiers can easily be fooled by the designed adversarial samples and get unexpected results. This phenomenon has raised serious concerns, especially for the security of applications based on deep learning [20–22]. The vulnerability of deep learning is first revealed in [23]. By adding adversarial perturbations to original images, we can obtain deceptive images imperceptive to human eyes. Subsequently, numerous attack methods were proposed. Goodfellow proposed a hypothesis that the existence of adversarial examples may be due to deep model linear property. Based on this hypothesis, they proposed a fast gradient sign method (FGSM) [24], which can modify the pixel values of images. Hence, the modified images can obtain the misclassify result. Later, an iterative version of FGSM, the basic iterative method (BIM) [25], was proposed. BIM can achieve an effective attack by using the generated more imperceptible adversarial perturbation. Then, projected gradient descent (PGD) [26] attack was proposed, which is seen as a variant of BIM. Unlike BIM, PGD adds a step of initializing the uniform random perturbation. Then, it can run more iterations until finding an adversarial example. Also, it replaces the clip operation of BIM on the gradient with the gradient projection [27]. Due to the above improvements, the attack capability of PGD is far superior to both FGSM and BIM. Moreover, PGD is known as the strongest first-order attack. Another strong attack is the Carlini and Wagner (C & W) [28] attack. C & W proposed a family of three attacks that minimize diverse similarity metrics: L_0, L_2, and L_∞, respectively. C & W achieves the effect that is imperceptive by humans, and they constrain the L_0, L_2, and L_∞ norm to make the perturbation small. Although C & W can generate more precise perturbations, the computational efficiency of C & W is slower. By contrast, PGD is a commonly used attack method that is stable and efficient at present.

The research on the adversarial samples for RSI scene classification is not thoroughly studied yet. Czaja et al. [29] provided a preliminary analysis of the adversarial examples for remote sensing data. However, their analysis is only under the targeted attack. Xu et al. [30] used FGSM and L-BFGS [23] attacks to generate the adversarial samples for RSI scene classification. They tested these above two attacks under the targeted and the untargeted attacks. We call it Xu2020 in this paper. Additionally, they utilized an adversarial training strategy to increase the resistibility of deep models that faced adversarial examples. Burnel et al. [31] proposed a different approach based on a modified Wasserstein generative adversarial network (GAN) [32] to generate natural adversarial hyperspectral images. Chen et al. [33] tested two adversarial attack algorithms (i.e., FGSM and BIM) on two classifiers trained on different remote sensing data sets. They found that a higher similarity in the feature space is more straightforward for generating the corresponding adversarial examples. Although the above methods can easily fool the classifier, these attack methods could not represent a state-of-the-art performance according to the attack capacity. FGSM attack has the limitation of its one-step characteristics. Hence, it has a low success rate on large public data sets. Unlike FGSM, BIM has lower iteration numbers than PGD, so its attack effect lags behind PGD.

Based on the above analysis, we utilize the PGD attack to generate adversarial samples. Then, we test the generated images on eight state-of-the-art classifiers to prove our method's effectiveness in this paper. The main contributions are summarized as follows:

(1) We apply a first-order attack (i.e., PGD) to generate the adversarial samples. This method can generate more accurate perturbation by uniform random perturbation as initialization, then running several BIM iterations to find an adversarial example

(2) To demonstrate our method's effectiveness, we employ three benchmark multicategory remote sensing scene data sets imaged by different sensors. Also, eight CNN-based classifiers are attacked under the untargeted and the targeted settings. Compared with the Xu2020 method, our method can make the attacked classifiers much lower classification accuracy. This phenomenon has revealed that the current state-of-the-art classifiers still have potential security risks in the multisource RSI scene classification task

(3) Furthermore, we also provide visual comparisons of the generated perturbations and the adversarial samples. For the more objective visual quality evaluation, the peak signal-to-noise ratio (PSNR) is employed as the evaluation index. The results indicate that our method can generate imperceptible adversarial samples and has a stronger attack ability for current state-of-the-art classifiers

The rest of this paper is organized as follows. Section 2 introduces the main strategies for generating adversarial examples and the detailed expression of the PGD attack under untargeted and targeted settings. Section 3 shows three RSI scene data sets used in this paper and the corresponding experimental results. Conclusions are summarized in Section 4.

2. Method

Adversarial attacks principally followed these two strategies. (1) We maximize the classifier's loss function under the L norm constraint by the small constant value. This operation will cause the classifier to misclassify. (2) Using the L norm to constrain the perturbation to be minor, the human eye's imperceptible images are generated. The overall procedure of our method is shown in Figure 1.

Given a set of test remote sensing images $X = \{x_i, \ldots, x_n\}$, $x_i \in [0, 255]^n$, with the corresponding labels $y \in Y_{i=1}^n$, a CNN classifier F: $X \longrightarrow y \in Y_i^n$ is introduced to map images to the corresponding labels. The adversarial attack aims to find a perturbation, which makes the target label $y_{\text{adv}} \neq y_{\text{ture}}$. Usually, we maximize the loss function of original data under constraint norm (i.e., $\|\delta\|_p$), and δ denotes the perturbation. Hence, the process of finding perturbation can be regarded as an inner maximization problem, and this procedure can be expressed as

$$\max_{\|\delta\|_p} \mathscr{L}(\theta, x', y), \tag{1}$$

where $\mathscr{L}(\theta, x', y)$ is the loss function. θ denotes the model's parameters and x' demotes the generated adversarial sample. To find a small perturbation for the given image, therefore, we minimize $\mathscr{D}(x, x')$ for the input image x, where $\mathscr{D}$ is the distance metric between the original image and the adversarial image. The procedure can be described as

$$\begin{aligned} &\text{minimize } \|\delta\|_p + c \cdot f(x'), \\ &\text{such that } x' \in [0, 1]. \end{aligned} \tag{2}$$

However, the optimization problem of equation (2) is difficult to be directly solved. Hence, FGSM was proposed, which is an efficient one-step attack. This attack used an L_∞-bounded constraint and the sign function to get its specific gradient direction. Then, it adjusts the input data by a small step in the direction that will maximize the loss until it gets the suitable perturbation. Finally, the perturbed image can be obtained:

$$x\prime = x + \varepsilon \operatorname{sign}\left(\nabla_x \mathscr{L}(x, y, \theta)\right), \tag{3}$$

where ε denotes a small constant value that restricts the perturbation. FGSM is based on a linear model, and the direction of loss is fixed. Even if we iterate it multiple times, the direction of the perturbation will not change. However, for a nonlinear model, the direction may not be completely correct after only one iteration. Hence, the multistep iterative version of FGSM was proposed, i.e., PGD attack. PGD uniforms the random perturbation as the initialization firstly. Then, an adversarial example is found by running several iterations of BIM. PGD creates a stronger attack than other previous iterative methods (e.g., BIM). Formally, the iterative procedure follows:

$$x_{i+1}{}' = \prod_{x+S} \left[x' + \alpha \operatorname{sign}\left(\nabla_x \mathscr{L}(x, y, \theta)\right)\right], \tag{4}$$

where Π denotes the projection operator, which clips the input at the positions around the predefined perturbation range. α means a gradient step size, and $x + S$ represents the perturbation set. $\mathscr{L}$ is the cross-entropy loss. Usually, L_∞ and L_2 norms are currently used as the constraint that bound the perturbation to be small. In this paper, [34] mentioned that L_∞ is the optimal distance metric for the image classification task, and [35] argues that the distillation is secure under this distance metric. Thus, we choose PGD-L_∞ as our attack method (Algorithm 1).

Generally, adversarial attacks can fool the CNN classifier under untargeted and targeted settings. The targeted attack means that the attacker assigns the classification result to a specific class. Unlike the targeted attack, the untargeted attack aims to make the classifier mispredict the given image's label but does not necessarily specify the prediction into a specific label. The targeted and the untargeted settings of our attack method can be formulated as

$$\begin{aligned} \text{target: } x_{\text{adv}}^{i+1} &= \prod_{x+s} \left[x_{\text{adv}}^i + \alpha \operatorname{sign}\left(\nabla_x J(x, \hat{y}, \theta)\right)\right], \\ \text{untarget: } x_{\text{adv}}^{i+1} &= \prod_{x+s} \left[x_{\text{adv}}^i + \alpha \operatorname{sign}\left(\nabla_x J(x, y, \theta)\right)\right]. \end{aligned} \tag{5}$$

3. Experiments and Results

3.1. Experimental Setup. We tested the Xu2020 method and our method on three RSI scene classification data sets, i.e., UC Merced (UCM) data set [10], WHU-RS19 data set [36], and AID data set [37]. Figures 2–4 show the examples of the three data sets.

UCM data set includes 2,100 remote sensing scene images. This data set consists of 21 scene classes, and each class contains 100 samples with a size of 256×256 pixels. All images were collected from the National Map Urban Area Imagery collection, including multiple towns across the United States. The 21 land-use classes are agricultural, airplane, baseball diamond, beach, buildings, chaparral, dense residential, forest, freeway, golf course, harbor, intersection, medium-density residential, mobile home park, overpass, parking lot, river, runway, sparse residential, storage tanks, and tennis courts. The WHU-RS19 data set was collected from Google Earth (Google Inc.). The WHU-RS19 has 19 different scene classes, and each class contains 50 images with the size of 600×600 pixels. The 19 scene classes are airport, beach, bridge, commercial area, desert, farmland, football field, forest, industrial area, meadow, mountain, park, parking lot, pond, port, railway station, residential area, river, and viaduct. AID data set is the biggest data set among the three data sets, containing 30 scene classes. Each class has around 220–420 images. This data set includes

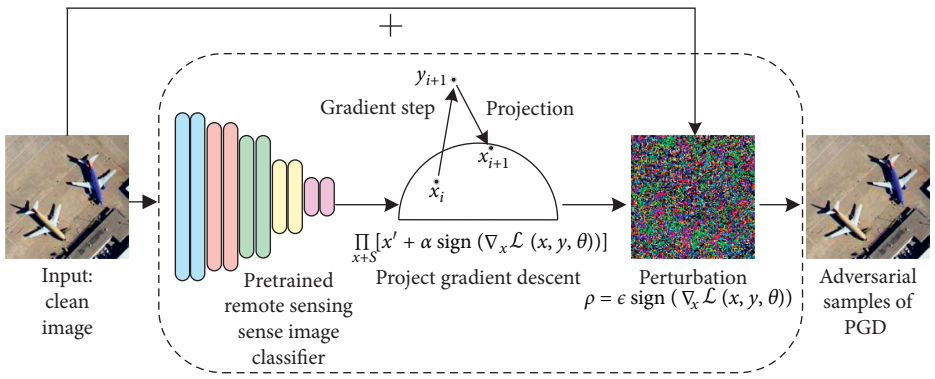


Figure 1: The overall procedure of our method.

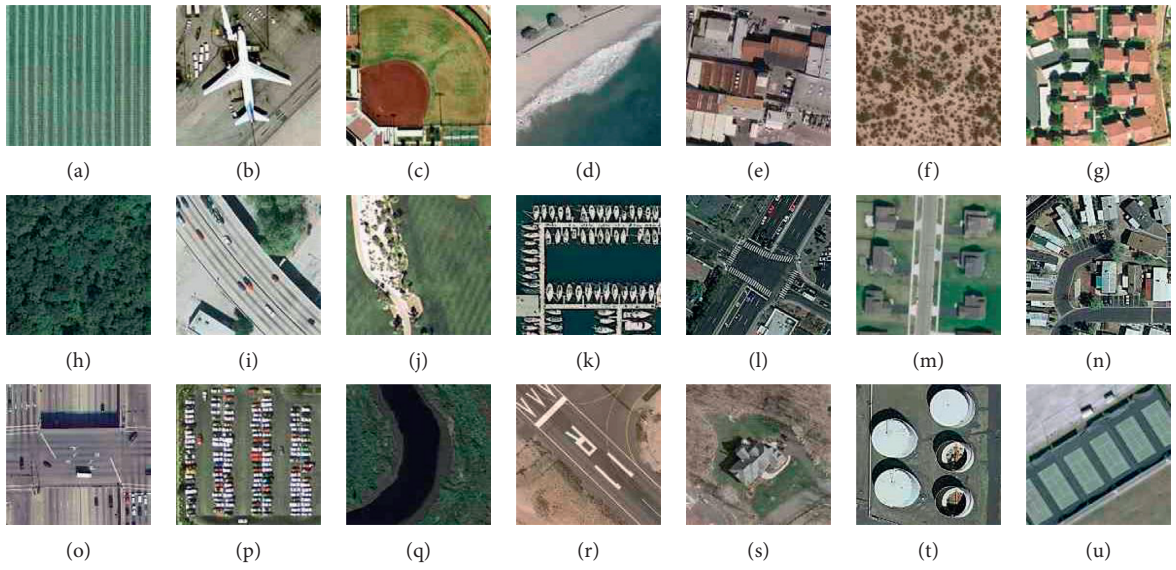


Figure 2: Examples of UCM data set. (a) Agricultural. (b) Airplane. (c) Baseball diamond. (d) Beach. (e) Buildings. (f) Chaparral. (g) Dense residential. (h) Forest. (i) Freeway. (j) Golf course. (k) Harbor. (l) Intersection. (m) Medium residential. (n) Mobile home park. (o) Overpass. (p) Parking lot. (q) River. (r) Runway. (s) Sparse residential. (t) Storage tanks. (u) Tennis court.

Input:
(1) A set of test remote sensing images $X = \{x_1, \ldots, x_n\}$ and the corresponding labels $Y = \{y_1, \ldots, y_n\}$.
(2) A classifier f with parameter θ.
(3) Number of epochs t; a small constant that restricts the perturbation parameter α;
Output: Adversarial example, x_{adv};
(1) Let $x_{a\ dv} = x_{i=1}^{n}$, $t = 0$,
(2) **while** iter $< t$ **do**
(3) Compute $\mathscr{L} = \sum_{i=1}^{n} \mathscr{L}(x, y, \theta)$
(4) Compute $x_{\text{adv}}^{i+1} = x_{\text{adv}}^{i} + t \times L$
(5) Compute perterbed = $x_{\text{adv}}^{i} - x_{i=1}^{n}$
(6) Compute perterbed = clip(perterbed, $-\varepsilon, \varepsilon$)
(7) $t+ = 1$
(8) **end while**
Return x_{adv}

Algorithm 1: Adversarial attack strategy.

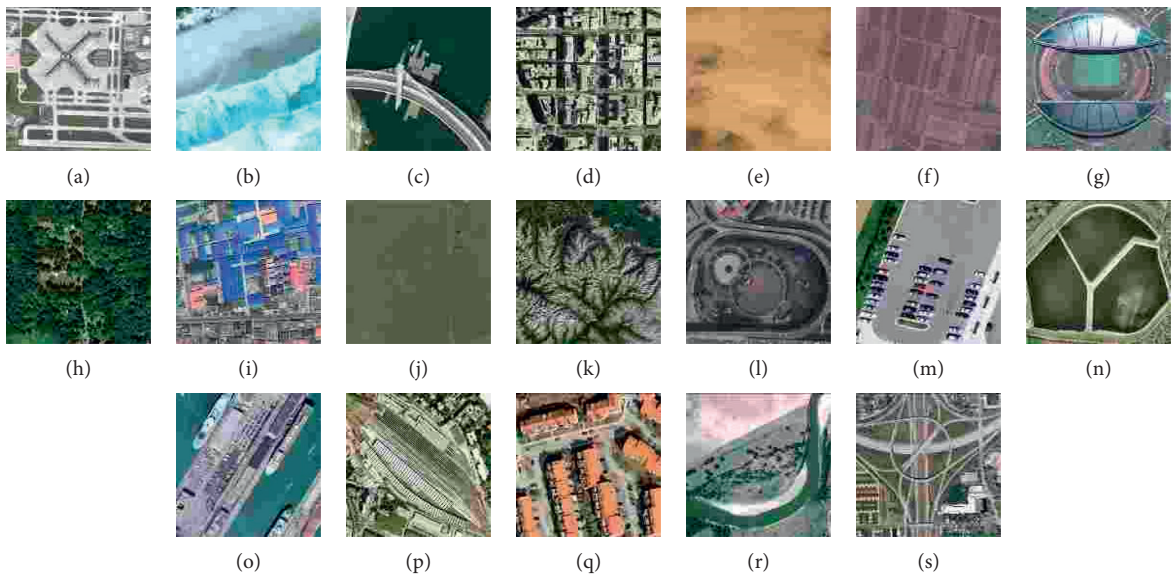


Figure 3: Examples of the WHU-RS19 data set. (a) Airport. (b) Beach. (c) Bridge. (d) Commercial. (e) Desert. (f) Farmland. (g) Football field. (h) Forest. (i) Industrial. (j) Meadow. (k) Mountain. (l) Park. (m) Parking. (n) Pond. (o) Port. (p) Railway station. (q) Residential. (r) River. (s) Viaduct.

10,000 remote sensing scene images. The 30 scene classes contain airport, bare land, baseball field, beach, bridge, center, church, commercial, dense residential, desert, farmland, forest, industrial, meadow, medium residential, mountain, park, parking, playground, pond, port, railway station, resort, river, school, sparse residential, square, stadium, storage tanks, and viaduct.

To prove the strong attack ability of our method, we conducted the experiments in three parts. Firstly, we compared the classification accuracies of the Xu2020 method (used in [30]) with those of our method. Both attacks were tested in the untargeted and targeted settings. Then, we also showed the classification accuracy specific to each class under the targeted setting. Finally, we visualized the generated adversarial images and the corresponding perturbations. For the more objective evaluation, we used the peak signal-to-noise ratio (PSNR) index to analyze the generated images of the Xu2020 and our method.

In this paper, we first randomly picked 20% of labeled images as the training set. The rest images are employed as the test set. All experiments were randomly repeated three times. In our experiment, we used Xu2020 as the compared method. Eight state-of-the-art CNN-based classifiers were employed to test our method. These classifiers were VGG-16 [38], GoogLeNet [39], InceptionV3 [40], ResNet-18 [41], ResNet-50 [41], ResNet-101 [41], DenseNet-121 [42], and DenseNet-201 [42], respectively. They were pretrained on the ImageNet [43] data set. Adam was used as the optimizer to train the classifiers with a batch size of 32. The training epochs were set as 40 with a learning rate initialized to $5e-5$ at the first 20 epochs and then the learning rate decay to $1e-5$ at the last 20 epochs. The experimental environment configuration is shown in Table 1.

3.2. Experimental Results of Untargeted Attack. In this section, we show the classification accuracies of eight different classifiers before and after the attack. The overall accuracy (OA, i.e., the number of accurately classified images divided by the number of entire test sets) is employed to evaluate different methods quantitatively. OA gap is the gap between the classification accuracy of the clean test set and that of the Xu2020 method and our method test set. For the untargeted and targeted attacks, the parameter ε was set to be 0.01 for the Xu2020 method and our method. Moreover, because Xu2020 is a one-step attack, we set the number of steps to 1. For our method, the number of steps was set to be 40. Tables 2–4 show the untargeted attack results against three types of input (clean, perturbed by Xu2020, perturbed by our method ran for 40 steps).

As shown in Table 2, we can see that all classifiers have excellent performance on three data sets before the attack. Generally, the OA values of eight classifiers are almost around 90%. Among these different classifiers, the classifiers with more complicated layers have higher accuracy than the classifiers with fewer layers. For instance, ResNet-101 has gained about 5% advancement in terms of OA than VGG-16 on the UCM data set. However, the OA of different classifiers all decreased dramatically after the attack.

As shown in Table 4, for the biggest data set (i.e., AID data set), the OA values of VGG-16 and DenseNet-201 are 20.97% and 53.88% after being attacked by Xu2020.

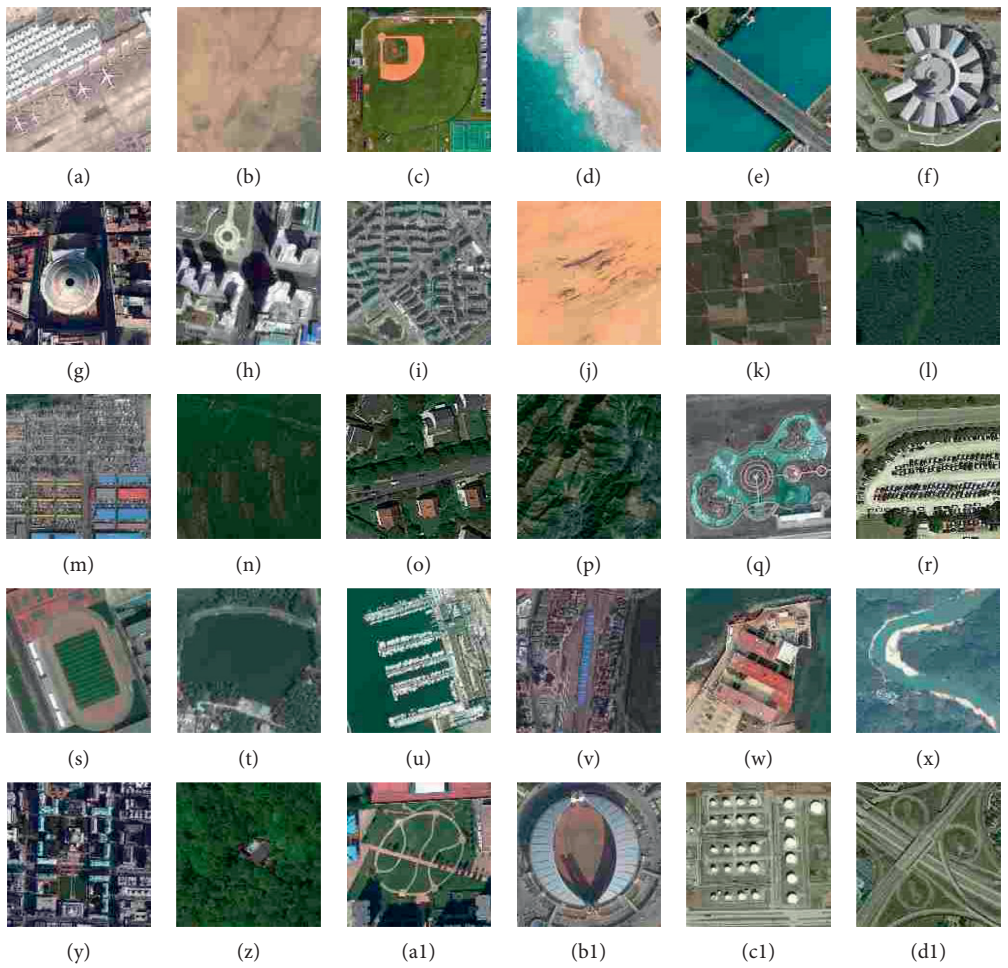


Figure 4: Examples of AID data set. (a) Airport. (b) Bare land. (c) Baseball field. (d) Beach. (e) Bridge. (f) Center. (g) Church. (h) Commercial. (i) Dense residential. (j) Desert. (k) Farmland. (l) Forest. (m) Industrial. (n) Meadow. (o) Medium residential. (p) Mountain. (q) Park. (r) Parking. (s) Playground. (t) Pond. (u) Port. (v) Railway station. (w) Resort (x) River. (y) School. (z) Sparse residential. (a1) Square. (b1) Stadium. (c1) Storage tanks. (d1) Viaduct.

Table 1: Experimental environment configuration.

Designation	Information
Operating system	Ubuntu 18.04 LTS
System configuration	CPU: Intel Core i7-9700 K 3.60 GHz, 32 GB RAM GPU: NVIDIA GeForce RTX 2080Ti 11G
Python library	Cuda 10.1 Pytorch 1.3.1 Torchvision 0.4.2 Numpy 1.19.2 Opencv-python 4.4.0 Advertorch 0.2.3

Table 2: Classification accuracy of the UCM data set under the untargeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	91.15 ± 1.03	24.46 ± 0.95	4.84 ± 0.73	−66.69	−86.31
GoogLeNet	95.52 ± 0.62	37.32 ± 1.73	5.93 ± 0.75	−58.19	−89.58
InceptionV3	94.78 ± 0.32	57.24 ± 1.03	10.00 ± 1.25	−37.54	−84.78
ResNet-18	94.80 ± 0.22	31.05 ± 1.33	1.92 ± 1.14	−63.75	−92.88
ResNet-50	96.27 ± 0.10	51.29 ± 1.27	2.60 ± 1.21	−44.98	−93.67
ResNet-101	96.15 ± 0.26	50.22 ± 4.27	2.66 ± 0.62	−45.93	−93.49
DenseNet-121	95.58 ± 0.67	50.08 ± 1.53	5.73 ± 1.51	−45.50	−89.84
DenseNet-201	95.91 ± 0.73	51.62 ± 0.68	5.65 ± 1.55	−44.29	−90.26

Table 3: Classification accuracy of the WHU-RS19 data set under the untargeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	91.03 ± 0.85	17.09 ± 2.79	0.24 ± 0.24	−73.94	−90.79
GoogLeNet	93.25 ± 1.54	31.52 ± 0.73	2.06 ± 0.48	−61.74	−91.19
InceptionV3	91.72 ± 0.57	45.90 ± 1.29	3.69 ± 0.88	−45.82	−88.03
ResNet-18	91.52 ± 1.21	22.42 ± 2.79	1.66 ± 0.44	−69.09	−89.86
ResNet-50	93.90 ± 0.44	41.05 ± 2.75	2.02 ± 1.41	−52.85	−91.88
ResNet-101	92.14 ± 0.87	42.95 ± 0.32	2.06 ± 0.00	−49.19	−90.08
DenseNet-121	93.05 ± 0.69	43.60 ± 1.62	3.64 ± 0.97	−49.45	−89.41
DenseNet-201	94.26 ± 0.53	51.52 ± 3.76	4.81 ± 1.25	−42.75	−89.45

Table 4: Classification accuracy of the AID data set under the untargeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	90.43 ± 0.27	20.97 ± 0.65	1.31 ± 0.74	−69.46	−89.12
GoogLeNet	91.97 ± 0.28	28.92 ± 0.44	0.42 ± 0.16	−63.05	−91.55
InceptionV3	92.26 ± 0.16	46.80 ± 0.44	4.49 ± 0.47	−45.46	−87.77
ResNet-18	92.22 ± 0.31	16.32 ± 1.13	0.15 ± 0.04	−75.90	−92.07
ResNet-50	93.54 ± 0.14	33.99 ± 1.08	0.23 ± 0.19	−59.55	−93.31
ResNet-101	93.80 ± 0.21	36.47 ± 1.93	0.34 ± 0.15	−57.34	−93.46
DenseNet-121	93.66 ± 0.15	42.93 ± 2.65	0.97 ± 0.12	−50.74	−92.69
DenseNet-201	94.14 ± 0.17	53.88 ± 1.19	2.89 ± 0.80	−40.25	−91.25

Table 5: Classification accuracy of the UCM data set under the targeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	91.15 ± 1.03	38.04 ± 0.38	29.98 ± 0.42	−53.11	−61.17
GoogLeNet	95.52 ± 0.62	51.21 ± 0.45	24.48 ± 0.97	−44.31	−71.04
InceptionV3	94.78 ± 0.32	66.09 ± 1.29	37.69 ± 1.95	−28.70	−57.10
ResNet-18	94.80 ± 0.22	42.89 ± 0.67	16.52 ± 0.70	−51.91	−78.28
ResNet-50	96.27 ± 0.10	60.84 ± 0.70	21.38 ± 0.19	−35.43	−74.89
ResNet-101	96.15 ± 0.26	60.93 ± 4.97	24.70 ± 2.02	−35.22	−71.45
DenseNet-121	95.58 ± 0.67	63.61 ± 1.23	26.14 ± 0.86	−31.97	−69.43
DenseNet-201	95.91 ± 0.73	64.17 ± 1.07	26.10 ± 0.63	−31.75	−69.81

Table 6: Classification accuracy of the WHU-RS19 data set under the targeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	91.03 ± 0.85	30.83 ± 2.30	26.61 ± 1.74	−60.20	−64.42
GoogLeNet	93.25 ± 1.54	47.80 ± 1.42	22.49 ± 0.76	−45.45	−70.76
InceptionV3	91.72 ± 0.57	57.02 ± 1.02	31.45 ± 0.75	−34.70	−60.27
ResNet-18	91.52 ± 1.21	37.46 ± 1.65	15.91 ± 1.14	−54.05	−75.60
ResNet-50	93.90 ± 0.44	51.64 ± 3.20	24.42 ± 2.19	−42.25	−69.48
ResNet-101	92.14 ± 0.87	54.72 ± 1.02	27.58 ± 1.65	−37.43	−64.56
DenseNet-121	93.05 ± 0.69	59.63 ± 1.66	28.05 ± 0.24	−33.42	−65.00
DenseNet-201	94.26 ± 0.53	65.18 ± 2.50	32.53 ± 0.64	−29.08	−61.73

Table 7: Classification accuracy of the AID data set under the targeted attack.

Models	OA of clean	OA of Xu2020	OA of ours	OA gap of Xu2020	OA gap of ours
VGG-16	90.43 ± 0.27	25.25 ± 0.41	22.55 ± 1.95	−65.18	−67.88
GoogLeNet	91.97 ± 0.28	44.08 ± 0.41	19.28 ± 0.74	−47.89	−72.69
InceptionV3	92.26 ± 0.16	63.56 ± 0.18	31.81 ± 0.49	−28.70	−60.45
ResNet-18	92.22 ± 0.31	29.12 ± 0.86	13.92 ± 0.38	−63.10	−78.30
ResNet-50	93.54 ± 0.14	50.03 ± 0.78	17.15 ± 0.41	−43.51	−76.39
ResNet-101	93.80 ± 0.21	53.76 ± 2.67	18.81 ± 0.74	−40.04	−75.00
DenseNet-121	93.66 ± 0.15	62.80 ± 1.69	24.33 ± 1.45	−30.87	−69.33
DenseNet-201	94.14 ± 0.17	74.03 ± 1.12	26.92 ± 1.84	−20.11	−67.21

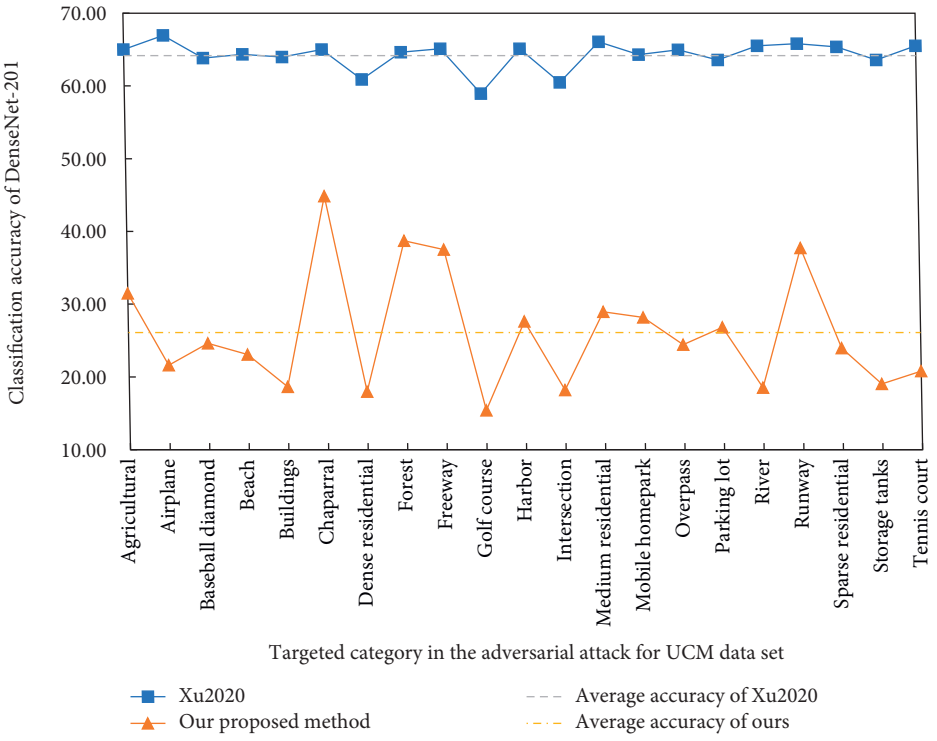


Figure 5: Each category classification accuracy of UCM under the targeted attacks of Xu2020 and our method.

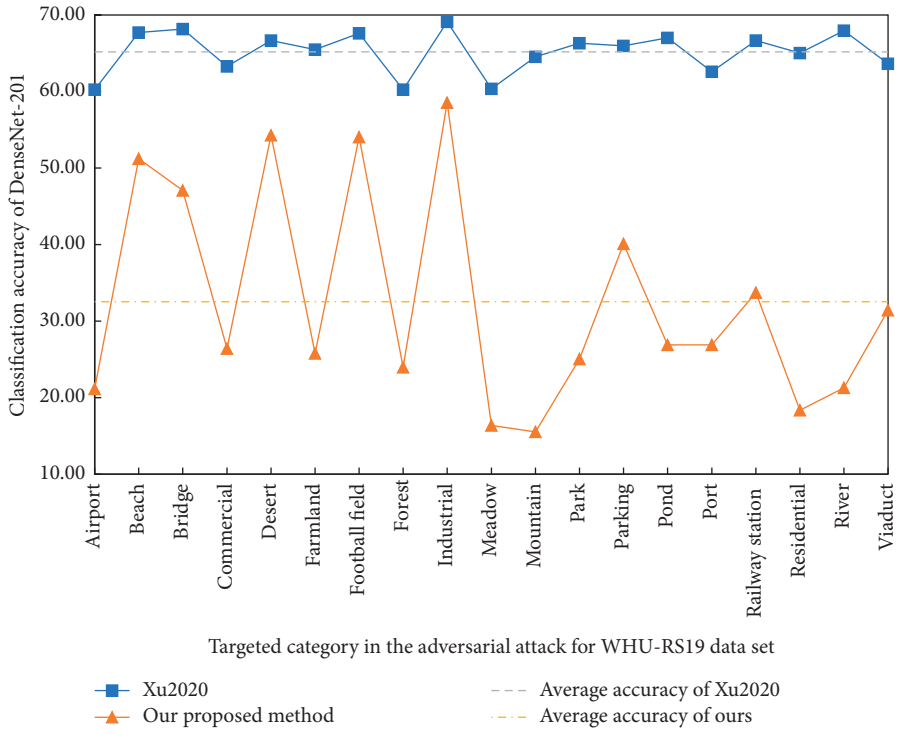


Figure 6: Each category classification accuracy of WHU-RS19 under the targeted attacks of Xu2020 and our method.

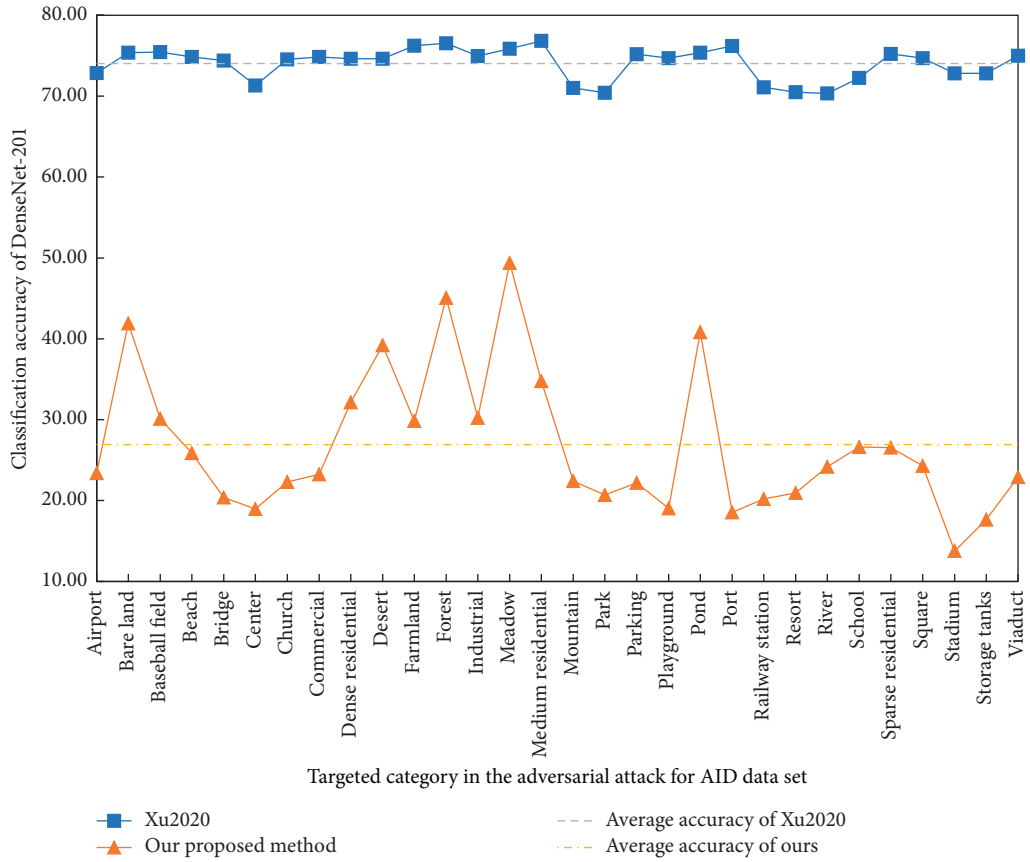


Figure 7: Each category classification accuracy of AID under the targeted attacks of Xu2020 and our method.

Meanwhile, they are 1.31% and 2.89% after being attacked by our method, respectively.

As shown in Tables 2–4, our method's adversarial images have the lower OA values for the bigger data set. Specifically, for the UCM data set (a small size data set with 21 classes), the OA value of DenseNet-201 is 5.65% after being attacked by our method. For the WHU-RS19 and the AID data sets, the OA values of DenseNet-201 are only 4.81% and 2.89% after being attacked by our method, respectively. These experimental results demonstrate the effectiveness of our method for different attacked classifiers and different data sets.

Also, we can find an interesting phenomenon from Tables 2–4. The OA gap between the clean and the adversarial test sets is highly related to the classifiers' depth for Xu2020. This phenomenon shows that the complicated classifiers tend to have a more robust and more stable property. However, the experimental results show that our method always has a stable OA gap value no matter the classifiers' depth. In summary, our method is not affected by the depth of the classifiers and can attack various classifiers stably.

3.3. Experimental Results of Targeted Attack. Furthermore, we also performed our method and the compared method under the targeted attack. Tables 5–7 represent the corresponding OA values.

In our experiments, we set the targeted attack categories as each category every time. The number of attacks was the number of the entire categories of the whole data set. The experimental results demonstrate that the classifiers also have vulnerability under the targeted attack. From Tables 5–7, we discover that the targeted attack's OA values are higher than those of the untargeted attack. For instance, for the RS19 data set, compared with the untargeted attack, the OA value has reached 26.61% on VGG-16 after being attacked by our method. However, the OA value has dropped to 0.24% on the same classifier under the untargeted attack. This phenomenon indicates that the targeted attack is more complicated than the untargeted attack.

We can find the common points from Tables 5–7. The OA gap between the shallow model (i.e., VGG-16) and the deep model (i.e., DenseNet-201) has a big difference. Specifically, for the AID date set under the targeted attack, the OA value is 25.25% on VGG-16 after being attacked by Xu2020. However, OA has reached about 74.03% on DenseNet-201. A similar phenomenon also can be found under the untargeted attack. The OA gap between VGG-16 and DenseNet-201 is around 30% on VGG-16. In contrast, our method shows a stable OA gap between the shallow and deep models. Specifically, under the targeted attack, the OA value is 22.55% on VGG-16 after being attacked by our method, and the OA value is 26.92% on DenseNet-201. The difference between shallow and deep models only is 4.37%.

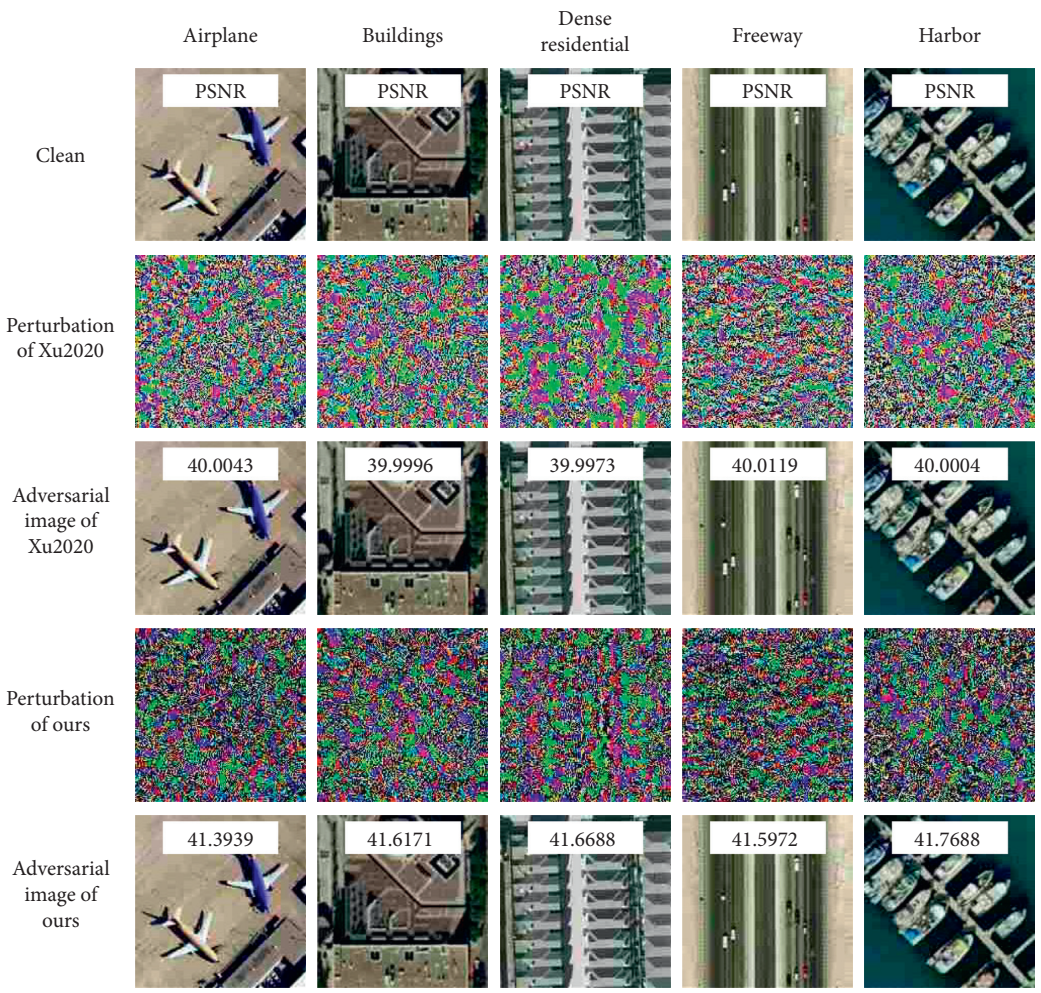


Figure 8: Untargeted adversarial samples generated by Xu2020/our method.

As shown in Tables 5–7, our method has no significant fluctuation between the OA gaps under targeted and untargeted attacks. This phenomenon proves that our method is more beneficial than Xu2020 under targeted and untargeted attacks.

To further analyze the adversarial attack's capacity, we provided each category classification accuracy (CA) after being attacked by Xu2020 and our method in Figures 5–7.

We can know that the CA values of Xu2020 are much lower than those of our method in each class. For instance, for the UCM data set, the CA value is 64.35% after being attacked by Xu2020. However, the CA value is 23.06% after being attacked by our method. As shown in Figures 5–7, the CA values are around 60% after being attacked by Xu2020 on each category in all three data sets. The gap between the highest and the lowest CA values was only 8.8% from Figure 5. However, the CA values have fluctuated after being attacked by our method. The possible reason is that the images' feature distributions of different categories have similar distributions.

An interesting phenomenon can be found in Figures 5–7. The CA values vary dramatically when the same classifiers are attacked by the same attack method but used different data sets. For the WHU-RS19 data set, the gap between the highest and the lowest CA values is 43.03% attacked by our method. However, for the bigger data set (UCM), the gap between the highest CA and the lowest CA values was only 29.46% when using our method. This phenomenon may be because the deep model is easily overfitting by training on a small data set.

3.4. Visualization of Adversarial Samples. In this section, we visualized adversarial samples generated by Xu2020 and our method. To further compare the generated images' quality, we also demonstrated the generated perturbations against the DenseNet-201 classifier on the UCM data set, as shown in Figure 8 (the untargeted attack) and Figure 9 (the targeted attack). These visualization results indicate that the

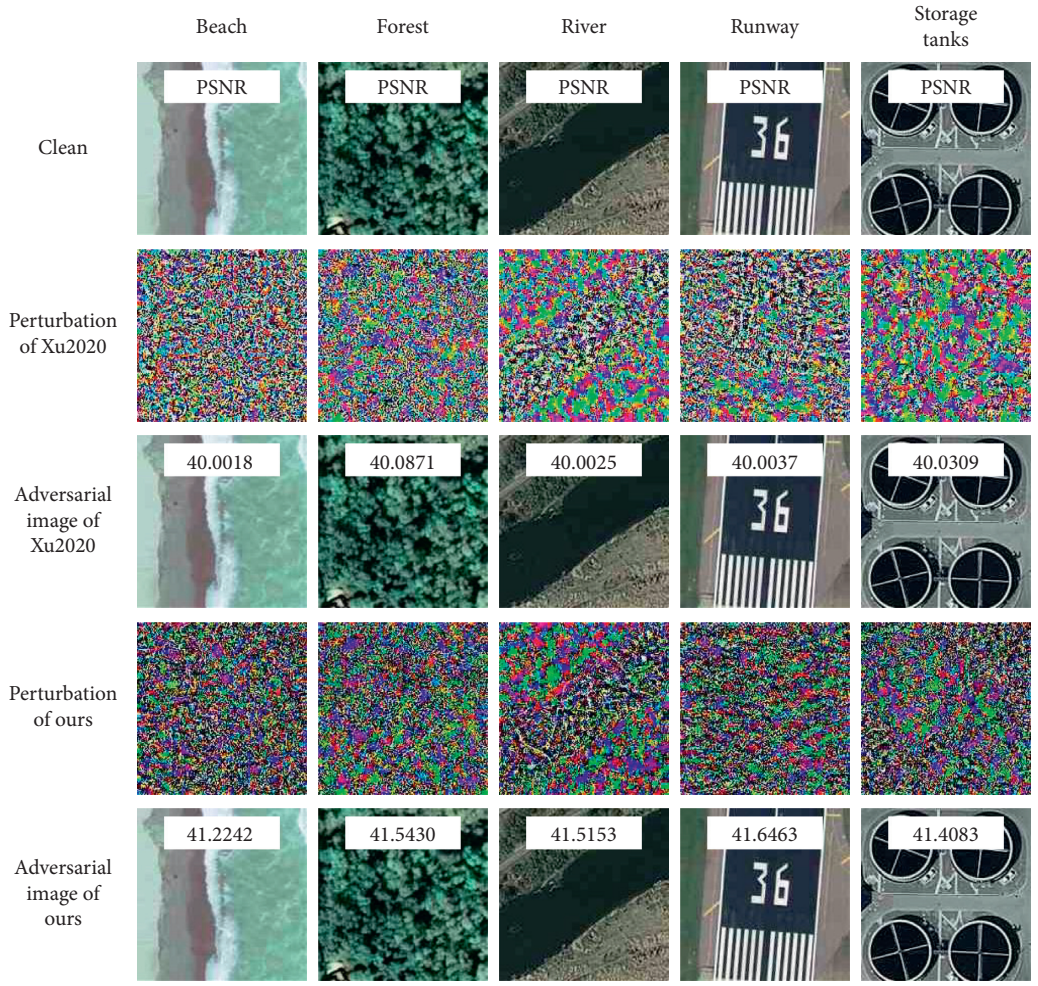


Figure 9: Targeted adversarial samples generated by Xu2020/our method.

perturbations generated by our method are more imperceptible to human eyes compared with Xu2020. The step of Xu2020 was set to 1, and the step of our method was set to be 40. ε is set to be 0.01. Under the targeted attack, we set the target category to "airport". The second and the fourth lines of Figures 8 and 9 represent the perturbations generated by the Xu2020 and our method, respectively. We can find that most of the perturbations generated by Xu2020 are more evident than those generated by our method, especially in the third and fifth perturbations of the second line under the targeted attack.

To evaluate visual quality more objectively, we employ the PSNR to compare the generated adversarial images, as shown in Figures 8 and 9. The PSNR is a common index, which can measure the similarity between the original clean and the adversarial images. The calculation of PSNR is given as

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{\text{MAX}_I^2}{\text{MSE}}\right), \tag{6}$$

where MAX_I represents the maximum possible pixel value of the image; MSE is the mean squared error. The higher value of PSNR indicates that the adversarial image's distribution is closer to the original clean image's distribution. If the PSNR value is higher than 40 dB, it means that the image quality is excellent. If the PSNR value is between 30 and 40 dB, it usually indicates that the image quality is good, which means that the distortion of the generated image is detectable but acceptable. The PSNR values are provided on the top of the third and fifth lines of Figures 8 and 9.

Figure 8 presents the adversarial images and perturbations generated by Xu2020 and our method. We can see that the PSNR values of Xu2020 are slightly lower than those of our method. Hence, our method's distribution of the adversarial images is closer to those of original clean images. This inference also can be confirmed from the generated perturbations in Figure 9. These results show that the images generated by our method have a better trade-off between the attack strength and the image quality.

In summary, our method can maintain its attack strength for the different classifiers and the models with different depths. Also, our method is more beneficial than Xu2020 under untargeted and targeted attacks. Moreover,

our method's quality of generated adversarial images has less distortion than Xu2020.

4. Conclusions

In this paper, we discussed the adversarial sample problem on the RSI scene classification task. Although the classifiers based on CNN have a state-of-the-art performance on the classification task, they have vulnerability toward adversarial samples. This may cause security risks for the RSI scene classification application. Moreover, our method performs better than the compared method under targeted and untargeted attacks. Specifically, our method has a stronger attack strength and better attack stability. We also analyzed the PSNR values of the generated adversarial images, and the results indicated that our method's adversarial images have better image quality.

Adversarial samples are a potential threat to the RSI scene classification task. We will further investigate the effective defense or detection strategies against the adversarial samples for RSI image classification in our future work.

Data Availability

The data used to support the findings of this study are available from the website given by corresponding papers.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the Fundamental Research Funds for the Central Universities (3072020CFQ0602, 3072020CF0604, 3072020CFP0601, and 3072021CF0609) and 2019 Industrial Internet Innovation and Development Engineering (KY1060020002 and KY10600200008).

References

[1] G. Cheng, J. Han, L. Guo, Z. Liu, S. Bu, and J. Ren, "Effective and efficient midlevel visual elements-oriented land-use classification using VHR remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 8, pp. 4238–4249, 2015.

[2] G. Cheng, J. Han, L. Guo et al., "Object detection in remote sensing imagery using a discriminatively trained mixture model," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 85, pp. 32–43, 2013.

[3] E. Actual, "Remote sensing image retrieval with global morphological texture descriptors," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 5, pp. 3023–3034, 2014.

[4] Y. Yang and S. Newsam, "Comparing SIFT descriptors and gabor texture features for classification of remote sensed imagery," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 1852–1855, San Diego, CA, USA, October 2008.

[5] X. Huang, L. Zhang, and L. Wang, "Evaluation of morphological texture features for mangrove forest mapping and species discrimination using multispectral IKONOS imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 6, no. 3, pp. 393–397, 2009.

[6] D. G. Lowe and G. David, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[7] S. Chen and Y. Tian, "Pyramid of spatial relatons for scene-level land use classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 1947–1957, 2015.

[8] G. Cheng, L. Guo, T. Zhao, J. Han, H. Li, and J. Fang, "Automatic landslide detection from remote-sensing imagery using a scene classification method based on BoVW and pLSA," *International Journal of Remote Sensing*, vol. 34, no. 1, pp. 45–59, 2013.

[9] H. Sridharan and A. Cheriyadat, "Bag of lines (BoL) for improved aerial scene representation," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 3, pp. 676–680, 2015.

[10] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 270–279, San Jose, CA, USA, November 2010.

[11] B. Luo, S. Jiang, and L. Zhang, "Indexing of remote sensing images with different resolutions by multiple features," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 4, pp. 1899–1912, 2013.

[12] M. Wang, X. Zhang, X. Niu, F. Wang, and X. Zhang, "Scene classification of high-resolution remotely sensed image based on ResNet," *Journal of Geovisualization and Spatial Analysis*, vol. 3, no. 2, pp. 1–9, 2019.

[13] W. Zhang, P. Tang, and L. Zhao, "Remote sensing image scene classification using CNN-CapsNet," *Remote Sensing*, vol. 11, no. 5, p. 494, 2019.

[14] J. Zhang, C. Lu, X. Li, H.-J. Kim, and J. Wang, "A full convolutional network based on DenseNet for remote sensing scene classification," *Mathematical Biosciences and Engineering*, vol. 16, no. 5, pp. 3345–3367, 2019.

[15] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[16] M. Li, Y. Sun, H. Lu, S. Maharjan, and Z. Tian, "Deep reinforcement learning for partially observable data poisoning attack in crowd sensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6266–6278, 2020.

[17] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious BOT-IOT traffic detection method in IOT network using machine learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2020.

[18] Y. Sun, Z. Tian, M. Li et al., "Honeypot identification in softwarized industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5542–5551, 2020.

[19] C. Luo, Z. Tan, G. Min et al., "A novel web attack detection system for internet of things via ensemble classification," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5810–5818, 2020.

[20] L. Jiang, K. Qiao, R. Qin et al., "Cycle-consistent adversarial GAN: the integration of adversarial attack and defense," *Security and Communication Networks*, vol. 2020, Article ID 3608173, 9 pages, 2020.

[21] F. Yu, L. Wang, X. Fang, and Y. Zhang, "The defense of adversarial example with conditional generative adversarial

networks," *Security and Communication Networks*, vol. 2020, Article ID 3932584, 12 pages, 2020.

[22] W. Elmasry, A. Akbulut, and A.H. Zaim, "Deep learning approaches for predictive masquerade detection," *Security and Communication Networks*, vol. 2018, Article ID 9327215, 24 pages, 2018.

[23] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Banff, Canada, April 2014.

[24] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[25] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, April 2017.

[26] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, Canada, April 2018.

[27] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial llogit ppairing," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Mountain View, CA, USA, March 2018.

[28] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, USA, May 2017.

[29] W. Czaja, N. Fendley, M. Pekala, C. Ratto, and I. J. Wang, "Adversarial examples in remote sensing," in *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 408–411, Seattle, WA, USA, November 2018.

[30] Y. Xu, B. Du, and L. Zhang, "Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: attacks and defenses," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 2, pp. 1604–1617, 2020.

[31] J. C. Burnel, K. Fatras, and N. Courty, "Generating natural adversarial hyperspectral examples with a modified wasserstein GAN," in *Proceedings of the Computer & Electronics Security Applications Rendezvous*, Rennes, France, November 2019.

[32] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 214–223, Sydney, Australia, August 2017.

[33] L. Chen, G. Zhu, Q. Li, and H. Li, "Adversarial example in remote sensing image recognition," 2019, http://arxiv.org/abs/1910.13222.

[34] D. Warde-Farley and I. Goodfellow, "Adversarial perturbations of deep neural networks," *Perturbations, Optimization, and Statistics*, pp. 311–342, MIT Press, Cambridge, MA, USA, 2016.

[35] N. Papernot and P. McDaniel, "On the effectiveness of defensive distillation," 2016, http://arxiv.org/abs/1607.05113.

[36] G. S. Xia, W. Yang, J. Delon et al., "Structural high-resolution satellite image indexing," in *Proceedings of the ISPRS Technical Commission VII Symposium on Advancing Remote Sensing Science*, pp. 298–303, Vienna, Austria, July 2010.

[37] G.-S. Xia, J. Hu, F. Hu et al., "AID: a benchmark data set for performance evaluation of aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3965–3981, 2017.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[39] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, Boston, MA, USA, June 2015.

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[42] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, Honolulu, HI, USA, July 2017.

[43] J. Deng, W. Dong, R. Socher et al., "ImageNet: a large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, FL, USA, June 2009.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9954957, 11 pages
https://doi.org/10.1155/2021/9954957

Research Article

EOM-NPOSESs: Emergency Ontology Model Based on Network Public Opinion Spread Elements

Guozhong Dong,[1] **Weizhe Zhang**,[1,2] **Haowen Tan**,[3] **Rahul Yadav**,[1] **and Shuaishuai Tan**[1]

[1]*The Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China*
[2]*School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China*
[3]*Department of Computer Engineering, Chosun University, Gwangju 61452, Republic of Korea*

Correspondence should be addressed to Weizhe Zhang; weizhe.zhang@pcl.ac.cn

Received 17 March 2021; Revised 14 April 2021; Accepted 24 May 2021; Published 12 June 2021

Academic Editor: Liguo Zhang

The construction of an emergency ontology model plays an important role in emergency management, which is an important basis for emergency public opinion management and decision-making. Integration of network public opinion spread elements into the emergency ontology model is crucial for realizing knowledge sharing in the field of emergency and public opinion responses. In this study, we crawl a large amount of emergency data from different data sources and construct an emergency dataset. Based on this dataset, we analyze the public opinion elements of emergencies and propose an emergency ontology model based on network public opinion spread elements (EOM-NPOSESs). Thereafter, we consider the coronavirus disease (COVID-19) emergency as an example to construct the EOM-NPOSESs. Finally, we design some strategies to realize rule reasoning and present the COVID-19 emergency application based on the constructed EOM-NPOSESs and the geographic information system platform. The results demonstrate that EOM-NPOSESs can not only describe the semantic relationship between emergencies and emergency elements but also perform semantic logical reasoning on different emergencies.

1. Introduction

With the rapid development of the social economy and due to socialist modernization, various social emergencies continue to occur and develop. Emergencies have had a tremendous impact on people's lives and are a threat to the normal order of public safety. Emergencies occur suddenly and can lead to a serious social harm. They can be divided into four categories: natural disasters, accident disasters, public health incidents, and social security incidents. Therefore, the public is often concerned and discusses emergency situations, which becomes the focus of public opinion. We are currently in a period of prominent contradictions and fragility of social structures. Facing an increasing number of emergencies, leading cadres at different levels have led to the accumulation of valuable practical experience. Emergencies are caused by several factors, and as such, decision makers must consider all these factors. Whether the government releases information pertaining to emergencies in a timely manner has become a critical factor for calming emergencies.

Research related to emergencies involves public opinion [1–3], network security [4–6], and data mining analysis [7–12]. Ontology models have been widely used in the fields of knowledge engineering and natural language processing. Ali et al. [12] proposed a fuzzy ontology to describe semantic knowledge and its relationships in the transportation domain. Fuzzy ontology-based semantic knowledge can improve deep learning techniques that extract more accurate aspects or features from the transportation text. Modeling domain knowledge using related technologies has become a research focus. To handle various emergencies effectively and reduce the impact of an emergency, it is necessary to perform good ontology modeling, ontology representation, and ontology reasoning for emergencies. Traditional emergency ontology models [13–20] are based on the static emergency concept, which cannot fully describe the

semantic relationship between emergencies and network public opinion spread elements. These emergency ontology models have significant shortcomings in knowledge fusion, knowledge reasoning, and knowledge questions and answers. Our previous research results [21, 22] demonstrate that the spread and diffusion of online public opinion in similar emergencies have similar burst patterns and key users. Integrating network public opinion spread elements into the emergency ontology model is of great significance in realizing knowledge sharing in the field of emergency and public opinion responses to emergencies.

In this study, emergency data crawled by web crawlers were used to construct an emergency dataset. We analyzed the public opinion elements of emergencies and proposed an emergency ontology model based on network public opinion spread elements (EOM-NPOSESs). Protégé was used to complete the modeling of the emergency ontology model. Coronavirus disease (COVID-19) emergencies were used as examples of emergency ontology models for experimental verification. The main contributions of this study are as follows:

(1) Network public opinion spread elements were integrated to define and describe the emergency ontology model. An EOM-NPOSESs was proposed.

(2) An emergency class and emergency class element construction method was proposed. Protégé was utilized to complete the emergency ontology modeling of COVID-19.

(3) Reasoning rules for COVID-19 emergencies were set based on EOM-NPOSESs. An emergency response prototype system with reasoning rules was developed to verify the applicability of the EOM-NPOSESs.

2. Related Works

According to their domain, emergency ontology models can be divided into two categories: general and specific. Research on general emergency ontology models is not limited to a specific type of emergency, making it more widely applicable. Specific domain emergency ontology models can be divided into different categories according to their domains and are often applicable only in those domains. Emergency ontology models can also be divided into single- and multilevel emergency ontology models.

In the domain of single-level semantic models and application research, Bitencourt et al. [13] presented an innovative ontological model for response protocols to fire emergencies in buildings. The conceptual model can be reused using other ontologies and systems related to the fire emergency domain. Several studies have adopted hierarchical methods to model emergencies using multilevel semantic models and applications. Wang et al. [14] proposed the emergency case ontology model comprising two parts: the upper ontology eABC and the application layer ontology. The former was obtained by extending the ABC ontology, whereas the latter from four aspects. Zhu and Liu [15] proposed an emergent event ontology model, which can be divided into three levels: upper-level event type, lower-level event type, and event instance. The model can describe the time, location, object, and relationship between event types. Ni et al. [16] constructed an emergency knowledge model based on the emergency response plan system in China. The model is divided into three levels: top-level, domain, and application ontologies. Knowledge sharing was achieved by constructing an emergency knowledge system. The focus of the abovementioned emergency studies was to construct a semantic model for emergencies. Liu et al. [17] proposed the concept of the event ontology pattern and its development method, which captures formally reoccurring models and reuses existing emergency ontology vocabulary. Tan et al. [18] proposed a generic ontology pattern for an emergency system model. A set of reasoning rules for emergency evolution, emergency solutions, and emergency resource utilization reasoning was proposed to conduct emergency knowledge reasoning based on the emergency ontology pattern. van Hage et al. [19] presented a simple event model to model events in various domains and provided examples from two use cases: historical emergencies and emergencies in the maritime safety and security domains. Gaur et al. [20] introduced empathy ontology to conceptualize core concepts concerning emergency management and planning of hazard crises.

Research on emergency ontology models has yielded good results, but some problems remain.

It is difficult to find the logical relationship and internal connection of online public opinion spread elements in different emergencies.

Further, although the existing emergency ontology models focus on constructing semantic models, they do not provide sufficient verification of the availability of the model and exploration of its applications, such as semantic reasoning, visual analysis, and semantic association in specific fields.

3. Emergency Ontology Model Based on Network Public Opinion Spread Elements

3.1. Related Definitions and Formal Descriptions

Definition 1. Emergencies occur suddenly and cause serious social harm, and emergency response measures for natural disasters, accident disasters, public health incidents, and social security incidents must be taken. We use e to represent an emergency; e is composed of six elements that can be represented by a six-tuple as follows:

$$e = \langle T, P, O, A, \mathrm{EA}, L \rangle, \tag{1}$$

where T denotes the time element, which represents the occurrence time of the emergency; P is the place/location element, which represents the location of the emergency; O is the object element, which represents the objects participating in the emergency, including key users and spread platform of online public opinion; A denotes the action element, which represents the action word that caused the emergency; EA denotes the emotional attitude element, which represents the emotional attitude of netizens

(positive, neutral, and negative); and L denotes the level element, including the impact level of the emergency and the level of network public opinion.

Definition 2. Emergency set includes a collection of emergencies with common or similar emergency elements.

$$ES = \{E_1, E_2, E_3, \ldots, E_i, \ldots\}, \quad (2)$$

where $E_i = \{e_1, e_2, \ldots, e_i, \ldots\}$, $i \in (T, P, O, A, \mathrm{EA}, L)$, and E_i represents a collection of emergencies that are specifically identical or similar in terms of emergency elements i.

Definition 3. Hierarchy of the emergency set: If there is a hierarchical classification relationship between emergency sets ES_1 and ES_2, ES_1 is a subset of ES_2, which is expressed by $R_{is-\mathrm{sub}}(ES_1, ES_2)$. Thus, ES_1 is referred to as the lower emergency of ES_2, whereas ES_2 is referred to as the upper emergency of ES_1. For instance, an infectious disease emergency is a subordinate emergency category of a public health emergency. It can be expressed as $R_{is-\mathrm{sub}}$ (infectious disease emergency, public health emergency).

Definition 4. Nonhierarchy of emergency: (1) Composition relationship: If the emergency e_1 comprises the emergency class e_2, the two emergencies can be defined as having a composition relationship, which can be expressed as R_{composed}. (2) Causal relationship: If the occurrence of an emergency e_1 leads to an emergency e_2, the two emergencies can be defined as having a causal relationship, which can be expressed as $R_{\mathrm{causality}}$. (3) Follow relationship: In a specific time range, if the occurrence of emergency e_1 follows the occurrence of emergency e_2, the two emergencies can be defined as having a follow relationship, which can be expressed as $R_{\mathrm{following}}$. (4) Concurrency relationship: Emergencies e_1 and e_2 occur simultaneously in a specific time range; thus, the two emergencies can be defined as having a concurrent relationship, which can be expressed as $R_{\mathrm{concurrency}}$.

Definition 5. Emergency ontology is a shared and objectively existing emergency model, defined as a five-tuple: EO = ⟨UES, LES, R, Rules, Individuals⟩. The upper emergency set (UES) has a hierarchical structure, whereas the lower emergency set (LES) has a nonhierarchical structure. Emergency relationship (R) refers to the relationship set between the emergency sets or emergencies. Rules refer to the inference rules, including the inference rules for emergencies and emergency relationships. Individuals represent the emergency instances. The structure of the EOM-NPOSESs is shown in Figure 1. The upper emergency set is a general classification structure, whereas the lower emergency set is a lattice structure formed by the emergency relationships.

3.2. Emergency Ontology Model Construction

3.2.1. Emergency Set Construction. The emergency set in the EOM-NPOSESs has a hierarchical structure formed using emergency classification and coding (National Standard of the People's Republic of China GB/T 35561-2017) (https://www.gb688.cn/bzgk/gb/newGbInfo?hcno=4C037B649CB13F556E00FA63A56528AB), which is divided into three categories. The partial hierarchical classification relationship between the emergency sets is shown in Figure 2.

Emergencies related to natural disasters include seven subcategories: meteorological disasters, floods and droughts, earthquake disasters, geological disasters, marine disasters, biological disasters, and forest and grassland fires.

Emergencies related to accidents include seven subcategories: wars and conflicts, safety production accidents, public facilities and equipment accidents, transportation accidents, environmental pollution accidents, ecological damage accidents, and tourism accidents.

Emergencies related to public health incidents include five subcategories: infectious epidemic incidents, mass disease transmission incidents, occupational hazard incidents, food safety incidents, and animal epidemics.

Emergencies related to social safety incidents include six subcategories: terrorist attacks, foreign-related emergencies, economic security incidents, social hotspot incidents with significant impact, large-scale mass incidents, and riot incidents.

The third layer comprises 80 subcategories. For example, the infectious disease emergencies include plague epidemics, cholera epidemics, pulmonary anthrax epidemics, infectious atypical pneumonia epidemics, human infections with highly pathogenic avian influenza, other epidemics of infectious diseases managed according to category A, epidemics of category B infectious diseases, epidemics of infectious diseases managed according to category B, new infectious diseases or the introduction of infectious diseases that have not yet been discovered, and outbreaks of infectious diseases that have been eliminated.

In addition to the hierarchical relationship of the emergency set, EOM-NPOSESs also include a nonhierarchical emergency relationship, such as composition, following, concurrency, and causality. As shown in Figure 1, the COVID-19 emergency is used as an example to construct a basic emergency relationship structure. After the construction of the COVID-19 nonhierarchical relationship, it is linked to the emergency set in "Other epidemics of infectious diseases managed by category A."

The hierarchy of the emergency set was constructed using an open-source neural hierarchical multilabel text classification toolkit [23] based on the text description of an emergency. It associates emergencies to the corresponding emergency set according to the classification tags of the emergency. The nonhierarchy of the emergency set was manually labeled for association analysis.

3.2.2. Emergency Element Construction. The time ontology from Web Ontology Language (OWL) official recommendations (https://www.w3.org/TR/2017/REC-owl-time-20171019) was used in this study. The ontology specification provides a standard vocabulary to describe the topological relationship of a certain instant or period, as well as the

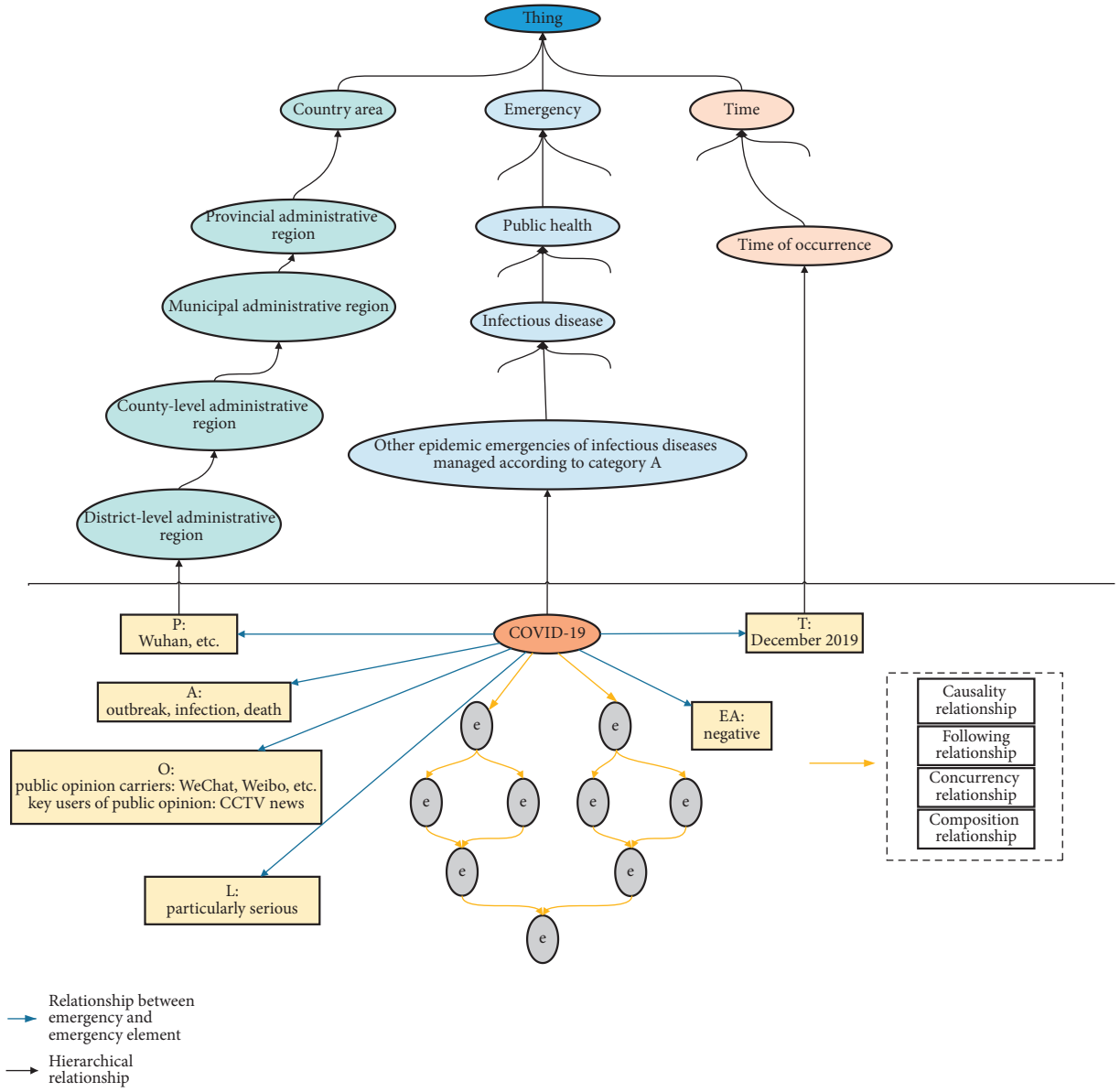


Figure 1: Structure of EOM-NPOSESs.

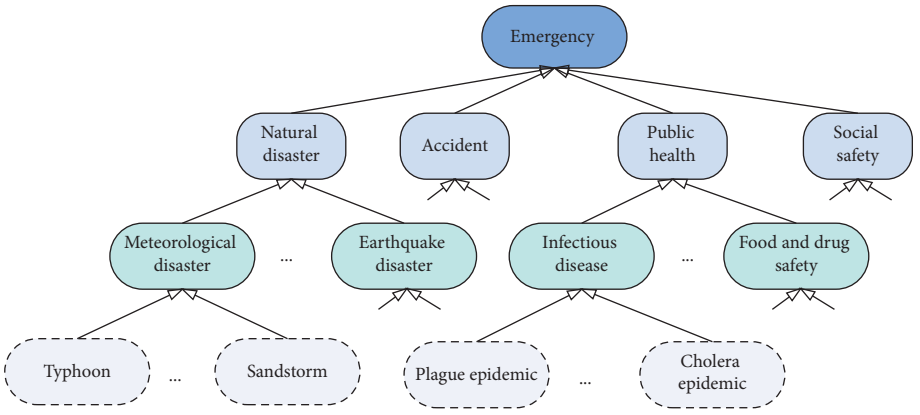


Figure 2: Partial hierarchical classification relationship in an emergency set.

duration and other information. The ontology also contains additional information related to date and time. In this study, the time element of emergency was taken as an instance of the corresponding time ontology model, and this element can be represented using the traditional calendar or other calendars as well, such as Unix-time or geological time.

According to the place name/address hierarchy rules and coding method in the national standard of the People's Republic of China-Digital City Geographic Information Public Platform Place Name/Address Coding Rules (GB/T 23705-2009) (http://www.gb688.cn/bzgk/gb/newGbInfo?hcno=DB5DF74A820991A754E40878CD3DA3B2), we extracted place/location element entities and constructed an ontology hierarchy of location elements. The ontology structure of the emergency place/location element in the EOM-NPOSESs is shown in Figure 3.

The object element of an emergency is a collection of objects participating in the emergency and in the dissemination of public opinion regarding the emergency, such as public opinion platforms and key users during the spread of online public opinion. Public opinion platforms include mainstream news portals and social media platforms (Weibo and WeChat). Key users are mined and analyzed based on the results of previous key user mining methods [21].

The action element of an emergency refers to the trigger word of an emergency [12]. Trigger words for emergencies can be extracted using named entity recognition.

The element of emotional attitude of an emergency describes the emotional attitude of the netizens (positive, neutral, and negative) toward that emergency. This element was analyzed using sentiment analysis.

The level of an emergency is determined based on the impact of the emergency, level of harm, level of network public opinion, relevant laws, and regulations.

3.2.3. OWL Description and Modeling of Emergency Ontology. This section presents the OWL modeling of emergency ontology through the extended OWL. The relationship between emergencies was established using the ObjectProperty type. Each specific emergency can be associated with other emergencies through several ObjectProperties. ObjectProperty is used to restrict the relationship between emergencies. There are three types of restrictions: "all values from," "some values from," and "has value." The restriction "all values from" implies that the specified attribute can only be obtained from the specified class; "some values from" implies that the specified attribute is obtained from the specified part of the class; and "has value" implies that the specified attribute can only take a fixed value. For instance, the tsunami emergency defines an ObjectProperty attribute that represents the following relationship: its restriction is some values from an earthquake, which represent tsunamis that occur with an earthquake.

In this study, the time element and place/location element of an emergency were considered as the subattributes of ObjectProperty. Further, the object, action, level, and emotional elements of an emergency were regarded as the subattributes of DataProperty. The lower emergency set in the ontology was associated with the corresponding upper emergency set with OWL: equivalentClass.

4. Experimental Evaluation

In this section, we first summarize the important emergencies and emergency elements obtained from certain emergency websites, news portals, Weibo, and WeChat. Thereafter, we compare the EOM-NPOSESs with other emergency ontology models. Finally, we consider the COVID-19 emergency as an example for constructing an ontology model. Combined with the ontology model, we also explain the design of some rule reasoning strategies and propose a COVID-19 emergency application based on EOM-NPOSESs using a geographic information system (GIS) platform.

4.1. Data Acquisition. In our experiments, the Qingbo public opinion big data platform (https://yuqing.gsdata.cn/) was used to collect COVID-19 public health emergency data based on related keywords. Its main data sources include news portals, Weibo, and WeChat. Simultaneously, a crawler program was compiled to crawl the China Emergency Service Network (https://www.52safety.com/), the National Emergency Broadcasting Network (https://www.cneb.gov.cn/), and Zhiwei (https://xgml.zhiweidata.net).

4.2. Experimental Results

4.2.1. Comparative Analysis of Emergency Ontology Models. Previous studies have constructed ontology models and explored applications for emergencies. The comparison results of the related emergency ontology models are summarized in Table 1. "Model" represents the name of model, "Domain" represents the application field of the ontology model, "General" means that the model can be applied to all types of emergencies, and "Specific" means that the model can only be applied to a specific emergency. "Hierarchical relationship" and "Nonhierarchical relationship" denote whether or not the hierarchical and nonhierarchical emergency relationships are defined in the model. "Emergency knowledge" and "Public opinion knowledge", respectively, represent whether or not the emergency and public opinion management-related knowledge are included in the emergency ontology model. Traditional emergency ontology models use a single-dimensional concept to indicate emergencies, which makes it difficult to incorporate emergency and public opinion knowledge. The emergency ontology model proposed in this study includes emergency knowledge, public opinion knowledge, as well as hierarchical and nonhierarchical relationships between emergencies, which can be applied to the fields of emergency management and public opinion response.

4.2.2. Emergency Ontology Model Analysis for COVID-19. Coronavirus disease (COVID-19) refers to the type of pneumonia caused by the 2019 new coronavirus infection. In December 2019, some hospitals in Wuhan discovered

Figure 3: Ontology structure of emergency place/location element in EOM-NPOSESs.

multiple unexplained pneumonia cases with a history of exposure to the South China Seafood Market. COVID-19 has now been confirmed as an acute respiratory infectious disease caused by the 2019 new coronavirus infection. The Director-General of the World Health Organization Tedros Adhanom Ghebreyesus held a press conference in Geneva to explain that the pneumonia epidemic caused by the new coronavirus constituted a public health emergency of international concern. The National Health Commission Director stated that COVID-19 is a major public emergency with the fastest spread, the widest range of infections, and the most difficult prevention since its founding. We used the COVID-19 emergency as an example to model the relationship between COVID-19 emergencies and the emergency elements to verify the validity of the emergency ontology model proposed in this study. Figure 4 shows the nonhierarchical relationships of the emergency ontology model for COVID-19. For convenience, only some emergencies are presented in Figure 4. Figure 5 shows the emergency knowledge representation of the emergency ontology model for COVID-19. Each emergency has a description of six elements: time, place/location, object, action, emotional attitude, and level.

4.2.3. Emergency Ontology Reasoning for COVID-19. Ontology reasoning is based on rules. Common reasoning engines can use grammatical structures to realize ontology reasoning. Jena's internal rule engine provides forward, backward, and hybrid models. In addition, custom inference rules can be used to construct hierarchical relationships and attribute corresponding axioms in the class diagram, such as expressing the relationship between characters and events in emergencies. Both the premise and conclusion comprise one or more atomic rules.

Figure 6 shows some key user elements of the emergency ontology model for COVID-19. Emergencies A (Tokyo Olympics faces cancellation of attention) and B (IOC allowed the postponement of the Tokyo Olympics) have the following relationship: If the network public opinion

Table 1: Comparison results of emergency ontology models.

Model	Domain	Hierarchical relationship	Nonhierarchical relationship	Emergency knowledge	Public opinion knowledge
EOP [17]	Universal	Y	Y	Y	N
Literature [18]	Universal	Y	Y	Y	N
SEM [19]	Universal	Y	N	N	N
Literature [15]	Universal	Y	Y	N	N
Empathi [20]	Specific	Y	N	Y	N
EOM-NPOSESs	Universal	Y	Y	Y	Y
EmergencyFire [13]	Specific	N	N	Y	N

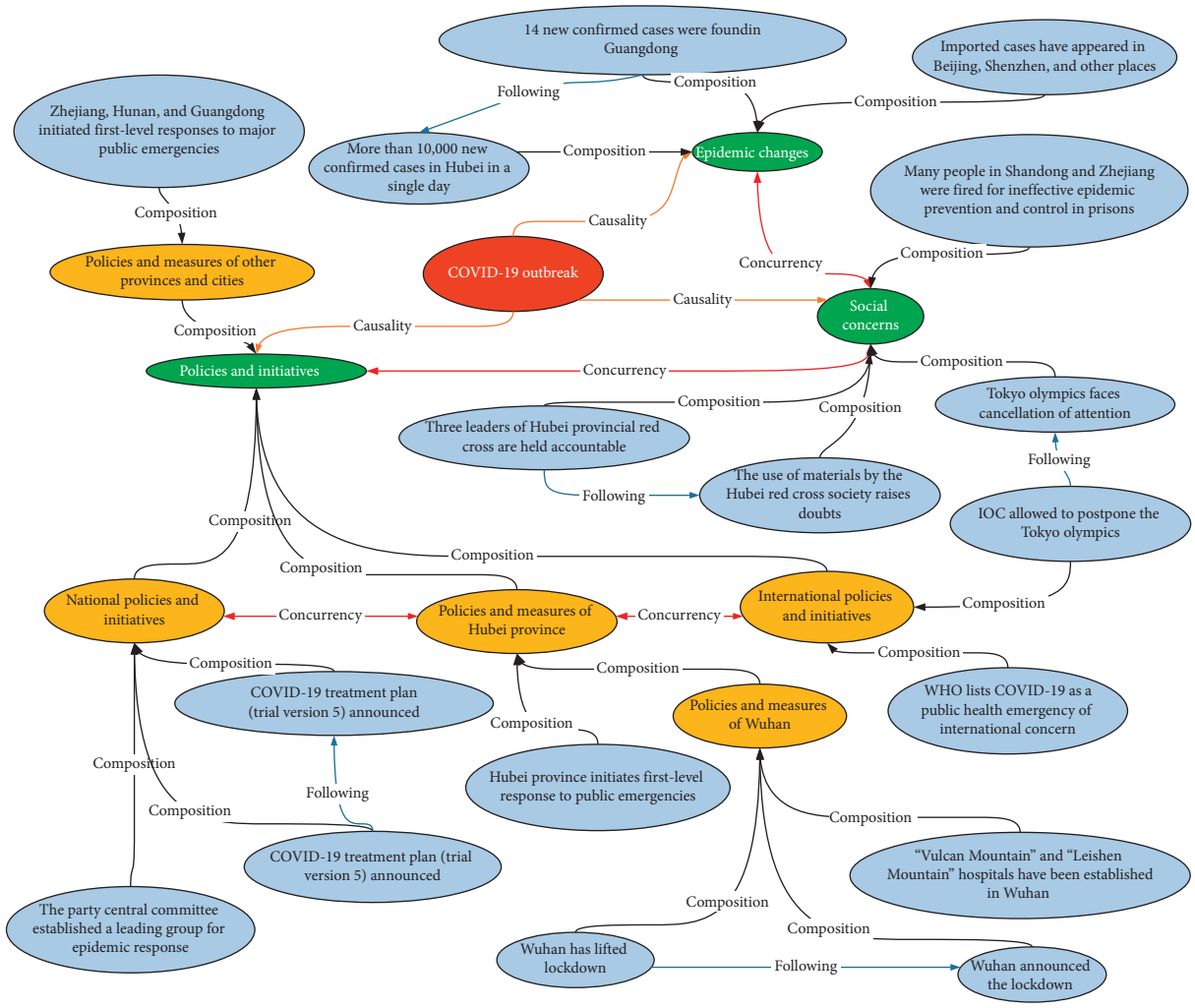


Figure 4: Nonhierarchical relationships of emergency ontology model for COVID-19.

dissemination platform and key users object element of Emergency B have been extracted, we can design key user inference rules for Emergency A. The key user inference rule is as follows.

[ruleHoldShare: (?) A: Following this relationship ?B) (? B: Key user ?c) ⟶ (?A: Key user ?c)]

As shown in Figure 6, headline news is the key user of Emergency B. If key users of Emergency A are not found, we can infer that the key user of event A is headline news. The reasoning results of the key users are consistent with those of the EOM-NPOSESs.

In addition, we verified other reasoning rules based on the EOM-NPOSESs. As shown in Figure 4, Emergency C (policies and measures of Hubei province) has a composition relationship with Emergency D (policies and measures of Wuhan city) and Emergency C has a

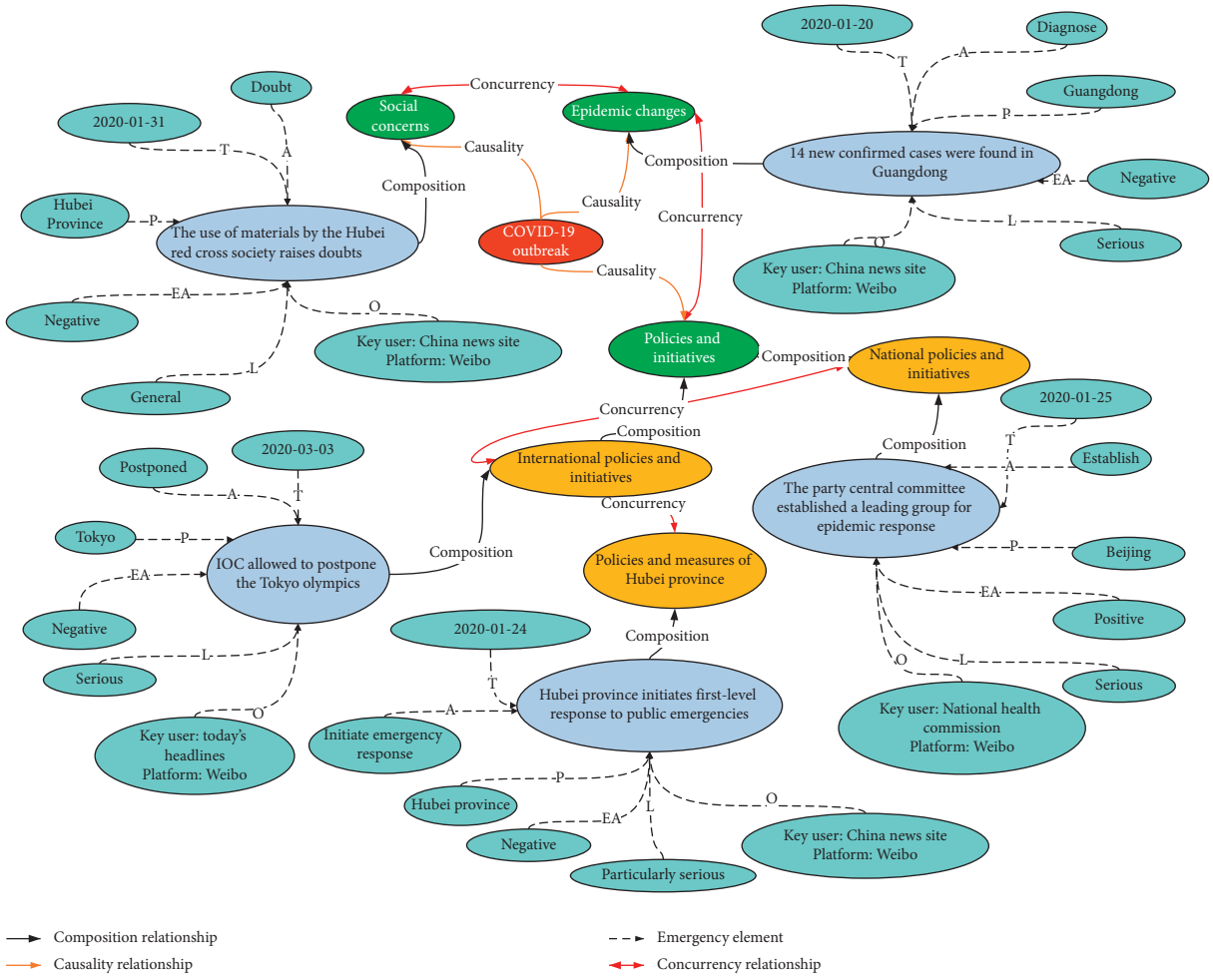


Figure 5: Emergency knowledge representation of emergency ontology model for COVID-19.

composition relationship with Emergency E (policies and measures of Huangshi city). Emergency D and the location element of Emergency E were extracted. As Wuhan and Huangshi cities are in Hubei province, we can refer to the policies and measures adopted in the Wuhan city to determine the epidemic policies and measures related to Emergency E. The customizable rules are as follows.

[ruleHoldShare: (?) a1: parallel relationship?a2) (?E: location element?a1) (?D: location element?a2)" + "(?c2: composition relationship,? D) ⟶ (?E: refer to ?c2)]

The ontology model proposed in this study presents the relationship between time, object, action, and the entire process of evolution of public opinion-related emergencies. Owing to the uncertainty of online public opinion and the development of emergencies, the reasoning results obtained using this model can provide a reference for emergency public opinion response and control departments. In addition, reasoning based on this model can help formulate emergency policies and measures for different emergencies. In other words, the emergency policies and measures selected by this model are helpful for guiding the development of an emergency response.

We combined our EOM-NPOSESs with a GIS platform to realize an emergency response prototype system. As shown in Figure 7, the developed system can dynamically display COVID-19 emergencies according to the timeline. In this system, the dynamic knowledge of an emergency is automatically instantiated, and dynamic knowledge reasoning and rapid knowledge support for emergency response actions are provided. To verify the role of this prototype system, we take the example of the COVID-19 emergency policies and measures in a specific city. The demonstration application of emergency policies and measures reasoning is shown in Figure 8. The reasoning results were found to be consistent with the actual results,

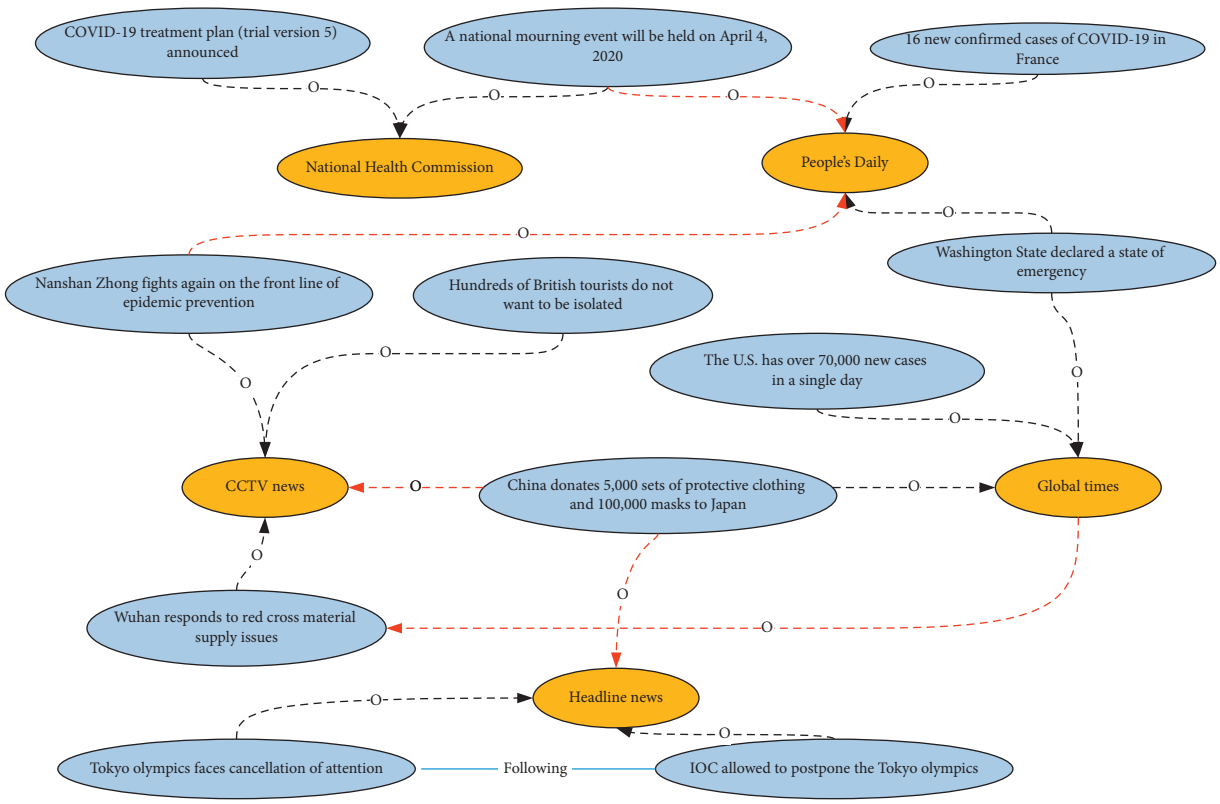


Figure 6: Some key user elements of emergency ontology model for COVID-19.

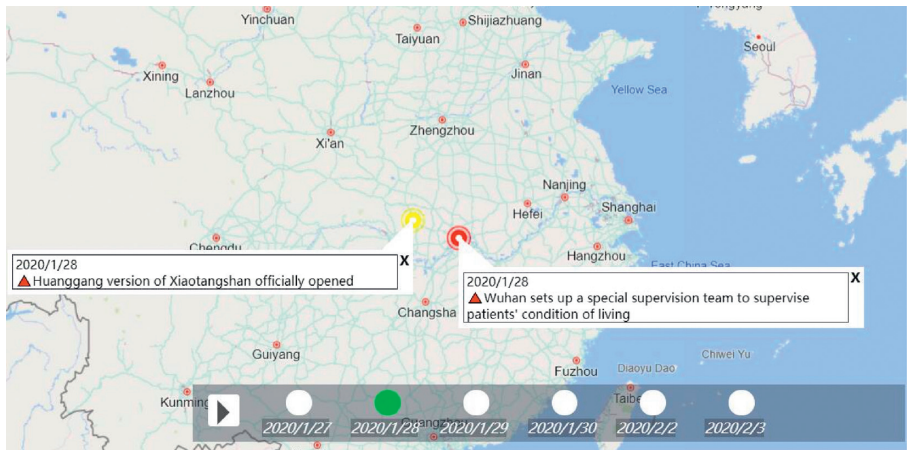


Figure 7: COVID-19 emergencies according to the timeline in the emergency response prototype system.

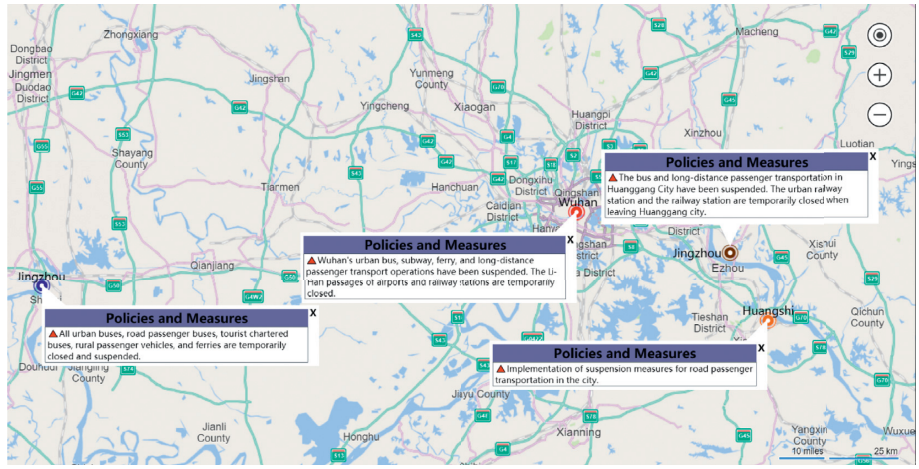


Figure 8: Demonstration application of emergency policies and measures reasoning.

and the usability of the EOM-NPOSESs model in practical applications was verified.

5. Conclusions

In this study, we constructed an emergency dataset and proposed EOM-NPOSESs. An emergency class and emergency class element construction method were proposed to construct the EOM-NPOSESs. Protégé was utilized to complete the OWL modeling of an emergency ontology through the extended OWL. Based on the constructed EOM-NPOSESs, the reasoning rules of COVID-19 emergency were set. We also developed an emergency response prototype system to verify the availability of the EOM-NPOSESs. The experimental results demonstrate that the EOM-NPOSESs can not only describe the semantic relationship between emergencies and elements of emergency but also perform semantic logical reasoning on different emergencies. In our future work, we will focus on extending our EOM-NPOSESs model and extracting the relations of emergency elements for multimedia data. We will also further investigate effective strategies for emergency relationship extraction.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publishing of the present study.

Acknowledgments

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province 2020B0101360001, and the Peng Cheng Laboratory Project (Grant no. PCL2021A02).

References

[1] C. You, D. Zhu, Y. Sun et al., "SNES: social-network-oriented public opinion monitoring platform based on ElasticSearch," *Computers, Materials & Continua*, vol. 61, no. 3, pp. 1271–1283, 2019.

[2] W. Yang, S. Rehman, and W. Que, "Identifying event-specific opinion leaders by local weighted LeaderRank," *Intelligent Automation & Soft Computing*, vol. 26, no. 6, pp. 1561–1574, 2020.

[3] G. Sun, S. Bin, M. Jiang et al., "Research on public opinion propagation model in social network based on blockchain," *Computers, Materials & Continua*, vol. 60, no. 3, pp. 1015–1027, 2019.

[4] F. Ali, A. Ali, M. Imran et al., "Traffic accident detection and condition analysis based on social networking data," *Accident Analysis & Prevention*, vol. 151, Article ID 105973, 2021.

[5] G. Dong, J. Gao, L. Huang, and C. Shi, "Online burst events detection oriented real-time microblog message stream," *Computers, Materials & Continua*, vol. 60, no. 1, pp. 213–225, 2019.

[6] J. Su, Z. Sheng, A. X. Liu, Y. Han, and Y. Chen, "Capture-aware identification of mobile RFID tags with unreliable channels," *IEEE Transactions on Mobile Computing*, pp. 1–141, 2020.

[7] S. H. Alsamhi, F. A. Almalki, O. Ma, M. S. Ansari, and M. C. Angelides, "Performance optimization of tethered balloon technology for public safety and emergency communications," *Telecommunication Systems*, vol. 75, no. 2, pp. 235–244, 2020.

[8] F. Ali, S. El-Sappagh, S. M. Riazul Islam et al., "An intelligent healthcare monitoring framework using wearable sensors and social networking data," *Future Generation Computer Systems*, vol. 114, pp. 23–43, 2019.

[9] S. Wang, Q. Peng, and H. Liang, "An event ontology model research for environmental pollution emergencies," *International Journal of Ambient Computing and Intelligence*, vol. 11, no. 4, pp. 38–56, 2020.

[10] M. Khan, P. Roy, I. Matin, M. Rabbani, and R. Chowdhury, "An adaptive governance and health system response for the COVID-19 emergency," *World Development*, vol. 137, Article ID 105213, 2021.

[11] H. Wang, H. Li, and H. Li, "Research of relation extraction method of civil aviation emergency domain ontology,"

Journal of Frontiers of Computer Science & Technology, vol. 14, no. 2, pp. 285–293, 2020.

[12] F. Ali, S. El-Sappagh, and D. Kwak, "Fuzzy ontology and LSTM-based text mining: a transportation network monitoring system for assisting travel," *Sensors*, vol. 19, no. 2, Article ID 234, 2019.

[13] K. Bitencourt, F. Durão, and M. Mendonça, "EmergencyFire: an ontology for fire emergency situations," in *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, pp. 73–76, Manaus, Brazil, October 2015.

[14] W.-j. Wang, P. Yang, and C.-x. Dong, "Study and application of emergency case ontology model," *Journal of Computer Applications*, vol. 29, no. 5, pp. 1437–1440, 2009.

[15] W. Zhu and Z. Liu, "Emergency domain knowledge modeling based on event ontology," *Computer Engineering and Application*, vol. 54, no. 21, pp. 148–155, 2018.

[16] Z.-j. Ni, L. Rong, N. Wang, and S. Cao, "Knowledge model for emergency response based on contingency planning system of China," *International Journal of Information Management*, vol. 46, pp. 10–22, 2019.

[17] W. Liu, Y. Tan, Y. Ding, Y. J. Zhang, and Z. T. Liu, "An ontology pattern for emergency event modeling," in *Proceedings of the 2016 IEEE 14th Intl Conference on Dependable, Autonomic and Secure Computing*, pp. 151–156, Auckland, New Zealand, August 2016.

[18] Y. Tan, W. Liu, Z. Yang, X. Du, and Z. Liu, "Pattern-based ontology modeling and reasoning for emergency system," *IEICE Transactions on Information and Systems*, vol. E101-D, no. 9, pp. 2323–2333, 2018.

[19] W. R. van Hage, V. Malaisé, R. Segers, L. Hollink, and G. Schreiber, "Design and use of the simple event model (SEM)," *Journal of Web Semantics*, vol. 9, no. 2, pp. 128–136, 2011.

[20] M. Gaur, S. Shekarpour, A. Gyrard, and A. Sheth, "Empathi: an ontology for emergency managing and planning about hazard crisis," in *Proceedings of the 13th IEEE International Conference on Semantic Computing*, pp. 396–403, Newport Beach, CA, USA, January 2018.

[21] G. Dong, W. Yang, F. Zhu, and W. Wang, "DPBT: a system for detecting pacemakers in burst topics," in *Proceedings of the 17th International Conference on Web-Age Information Management*, pp. 537–540, Nanchang, China, June 2016.

[22] G. Dong, W. Yang, F. Zhu, and W. Wang, "Discovering burst patterns of burst topic in twitter," *Computers & Electrical Engineering*, vol. 58, no. 2, pp. 551–559, 2017.

[23] L. Liu, F. Mu, P. Li et al., "NeuralClassifier: an open-source neural hierarchical multi-label text classification toolkit," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 87–92, Florence, Italy, January 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6625688, 10 pages
https://doi.org/10.1155/2021/6625688

Research Article

R2AU-Net: Attention Recurrent Residual Convolutional Neural Network for Multimodal Medical Image Segmentation

Qiang Zuo, Songyu Chen, and Zhifang Wang

Electronic Engineering College, Heilongjiang University, Harbin 150000, China

Correspondence should be addressed to Zhifang Wang; wangzhifang@hlju.edu.cn

Received 31 December 2020; Accepted 26 May 2021; Published 10 June 2021

Academic Editor: Liguo Zhang

In recent years, semantic segmentation method based on deep learning provides advanced performance in medical image segmentation. As one of the typical segmentation networks, U-Net is successfully applied to multimodal medical image segmentation. A recurrent residual convolutional neural network with attention gate connection (R2AU-Net) based on U-Net is proposed in this paper. It enhances the capability of integrating contextual information by replacing basic convolutional units in U-Net by recurrent residual convolutional units. Furthermore, R2AU-Net adopts attention gates instead of the original skip connection. In this paper, the experiments are performed on three multimodal datasets: ISIC 2018, DRIVE, and public dataset used in LUNA and the Kaggle Data Science Bowl 2017. Experimental results show that R2AU-Net achieves much better performance than other improved U-Net algorithms for multimodal medical image segmentation.

1. Introduction

Medical image plays a key role in medical treatment. Computer-aided diagnosis (CAD) is designed to provide doctors with accurate interpretation of medical images systematically so as to treat the patients better. Manual segmentation not only relies heavily on doctors' own knowledge and clinical experience in recognition accuracy, but also has very low efficiency. Therefore, the application of deep learning in medical image segmentation has aroused widespread concern. Because medical image labeling requires the experts to spend considerable time and effort, it is difficult to acquire thousands of training images in medical image segmentation tasks. Ciresan et al. [1] trained networks in sliding windows to predict class tags for each pixel by providing local areas (patch) around pixels. However, this network must run networks independently for each patch, and there are plenty of redundancies due to overlapping patches. Furthermore, more maximum pool layers are needed for large patches, which will reduce the positioning accuracy. Full convolutional neural network [2] is one of the earliest applied deep neural networks with image segmentation. Without traditional full connection layer, it uses deconvolution to restore original images at the last layer of network. Ronneberger et al. [3] extended this system and proposed U-Net, which includes coding path and decoding path. Encoder uses output feature map to characterize original image. Through the information output from encoder, decoder restores details and size of the original image. U-Net adds multiple skip connections between the encoder and decoder, which can transfer the features of the shallow network to the deep network. Thus, it can help the decoding path recover the details of the image better. Since then, U-Net becomes a very popular segmentation network and is applied to medical imaging segmentation including cardiac MRI [4], cardiac CT [5], abdominal CT [6] segmentation and pulmonary nodule detection [7], and liver segmentation [8]. However, target organs vary greatly among different patients, so U-Net will rely extremely on multicascaded CNN. Cascade framework will make dense predictions of ROI, which will lead to repetitive extraction of similar low-level features and result in the waste of computational resource and the increase in model parameters. Therefore, the design of an efficient structure of deep CNN is very important.

So far, many improved versions of U-Net have been proposed. Azad et al. [9] proposed BCDU-Net. The most important changes are for feature extraction method and skipping connections. The original U-Net relies on multi-cascaded CNN, which results in the waste of computing resources and the increase of the number of parameters. U-Net is able to splice shallow features and deep features simply by using skip connection. In this paper, an extended version of U-Net is proposed, which uses recurrent residual convolutional neural networks with attention gate connection (R2AU-Net) for medical image segmentation. The contributions of this paper can be summarized as follows: firstly, R2AU-Net uses more attention gates (AGs) to deal with deep features and shallow features. The AGs use the depth feature map in the decoding path as a gating signal to modify the feature map generated in the coding process and suppress feature responses in the irrelevant background area, so as to highlight features that are useful for a specific task [10]. Secondly, R2AU-Net substitutes recurrent residual convolutional unit for the U-Net basic convolutional unit. In the recurrent residual convolutional unit, recurrent connection and residual connection [11] are added to each convolutional layer [12], thus not increasing network parameters. The use of recurrent connection can enhance the ability of integrating context information. Residual connection can help train deeper network [13, 14]. In addition, batch normalization [15] is used to accelerate the convergence speed of the network. R2AU-Net is evaluated on three datasets: retinal vascular segmentation (DRIVE dataset), skin lesion segmentation (ISIC 2018 dataset), and lung nodule segmentation (lung dataset).

2. Proposed Method

Multiple deep learning models are usually taken as functional modules to construct the new network. Inspired by U-Net, R2AU-Net is proposed in this paper. The network structure is shown in Figure 1, which takes advantage of four recently developed deep learning models. There are three differences between R2AU-Net and U-Net. Recurrent convolutional block with the residual unit is used in encoding and decoding paths. Secondly, the skipping connections are replaced by AGs to correct low-resolution features through deep features. The third point is that BN [15] is used to increase the stability of the neural network and speed up the convergence speed of the network in the upsampling process. BN can standardize data, obtain smaller regularization, reduce generalization error, and improve network performance [15].

2.1. Encoding Path. The encoding path of R2AU-Net contains four steps. Every step contains a recurrent residual convolutional unit, which consists of two 3×3 convolution and adds recurrent connections to each convolutional layer to enhance the capability of integrating contextual information of the model. In addition, residual connections are added to develop more efficient and deeper models. Each time a recursive residual convolutional unit is passed, the number of feature maps is doubled and the size becomes half of the original. The R2AU-Net model applies concatenation on feature mapping from encoding unit to decoding unit. The recurrent convolutional layers (RCL) in R2CL are performed according to the discrete time steps expressed by RCNN [12]. Suppose the u_l input sample in the l^{th} layer of the R2CL block and a pixel located at (i, j) in an input sample on the k^{th} feature map in the RCL. $M_{ijk}^{l}(t)$ denotes the output at time step t and can be expressed as follows:

$$M_{ijk}^{l}(t) = \left(w_k^f\right)^T u_l^{f(i,j)}(t) + \left(w_k^r\right)^T u_l^{r(i,j)}(t-1) + b_k. \quad (1)$$

In formula (1), $u_l^{f(i,j)}(t)$ and $u_l^{r(i,j)}(t-1)$ represent standard convolutional layers and the input sample of the l^{th} RCL, respectively. The standard convolutional layer and the RCL of the k^{th} feature maps are, respectively, weighted by w_k^f and w_k^r, and b_k is the bias. The output of RCL is activated by standard ReLU function f as follows:

$$F(u_l, w_l) = f\left(M_{ijk}^{l}(t)\right) = \left(0, M_{ijk}^{l}(t)\right). \quad (2)$$

The output of the R2CL unit can be calculated as follows:

$$u_{l+1} = u_l + F(u_l, w_l), \quad (3)$$

where u_l is an input sample of R2CL layer. u_{l+1} is both the output of downsampling layer in encoding path and the output of upsampling layer in decoding path, respectively. The basic unit of U-Net convolution is shown in Figure 2(a), and the structure of the R2CL block is shown in Figure 2(b).

Formulas (1) and (2) describe the dynamic characteristics of RCL. When RCL is expanded into T time steps, the feedforward subnetwork with depth of $T+1$ will be obtained. In this paper, RCL is expanded to two time steps; namely, $T = 2$. RCL includes a single convolutional layer and two subsequence recurrent convolutional layers. RCL expansion structure is shown in Figure 3.

2.2. Decoding Path. Each step of the decoding path performs the upsampling operation of the output from the R2CL unit of the previous layer. With each upsampling operation, the number of feature maps will be halved and the size will be doubled. At the last layer of the decoding path, the size of the feature map is restored to the original size of the input image. The LNR layer in R2CL is replaced by BN layer, so that the input of each layer keeps the same distribution. In the process of training, the distribution of activation in each layer of neural network will lead to the decrease of training speed. Therefore, BN [15] is used to enhance stability of neural networks after sampling at each step. It improves the stability of the neural network by subtracting the batch mean and dividing the inputs according to the batch standard deviation. BN accelerates training speed and promotes the performance of network model.

The output u_l^{up} of BN layer is sent to AGs. R2AU-Net uses AGs to readjust the output features of the encoder before splicing the features on each resolution of the encoder with the corresponding features in the decoder. This module generates a gating signal which controls the importance of

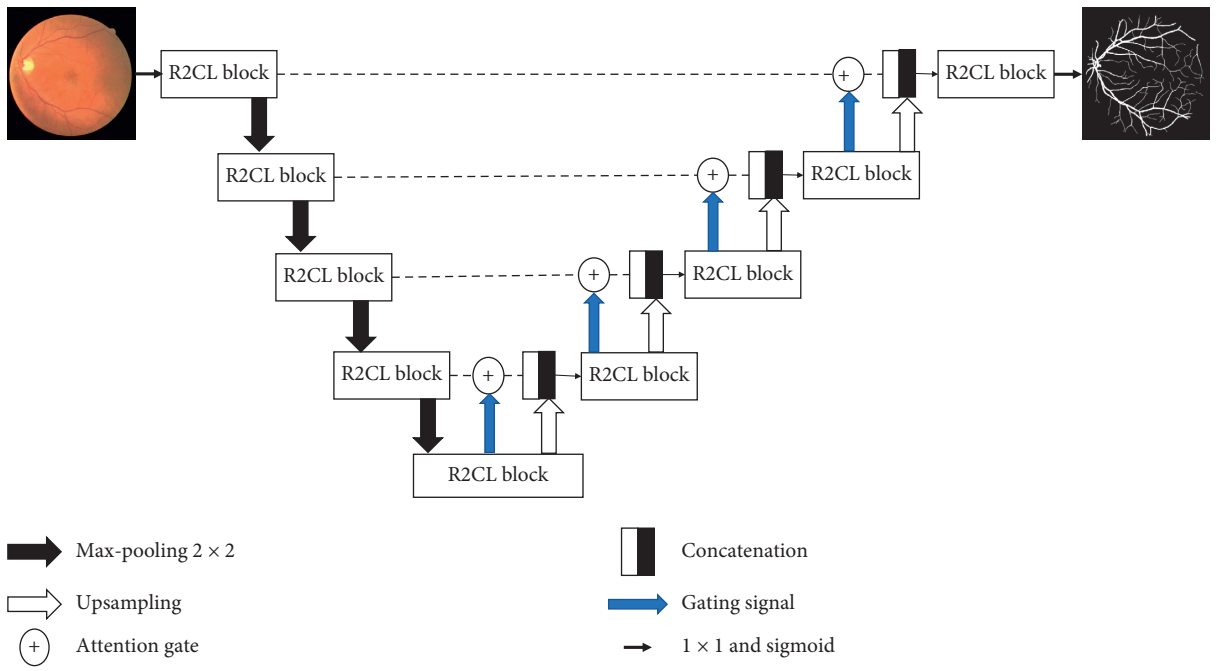


Figure 1: Convolutional encoding and decoding unit based on the recurrent residual convolutional layer (R2CL) and R2AU-Net structure with AG connection.

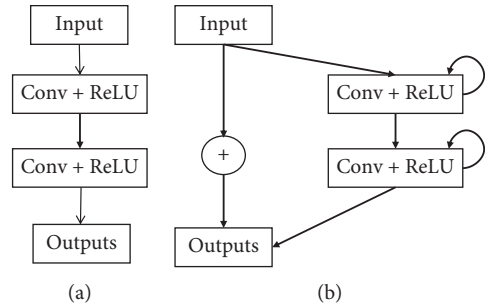


Figure 2: Two kinds of convolutional block. (a) The basic unit of the U-Net convolution. (b) R2CL block.

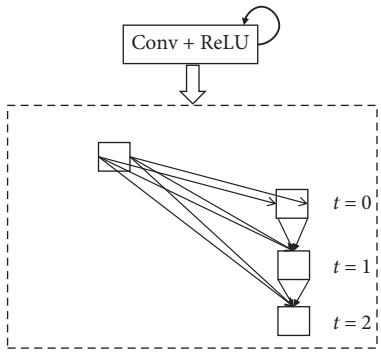


Figure 3: Unfolded recurrent convolutional units for $T = 2$.

features at different locations. AGs gradually suppress feature responses unrelated to background regions without clipping ROI regions between networks.

Figure 4 shows the proposed additive attention map. Attention values are calculated for each pixel $u_l^{f(i,j)}$, respectively. For the convenience of representation and distinction, u_l^{down} and $\hat{u}_l^{\text{up}}$ are denoted as u_l and g_l, respectively. The gating signal g_l determines the focus area per pixel. In order to obtain higher accuracy, the additive attention [16] is used to obtain the attention coefficient. The additive formula is as follows:

$$Q_L = \Phi\left(\sigma_1\left(W_u u_l + W_g g_l + b_g\right)\right) + b_\Phi, \quad \alpha_l = \sigma_2\left(Q_L\right), \tag{4}$$

where σ_1 and σ_2 denote ReLU and sigmoid activation functions, respectively, W_g is the weight, and b_g and b_Φ are the bias. Wang et al. [17] used attention based on vector splicing. Linear transformations are calculated using a $1 \times 1 \times 1$ convolution of tensor channel direction. Grid resampling of attention coefficients is achieved using trilinear interpolation. The update of AG parameter is trained according to backpropagation instead of using sampling-based update method [18].

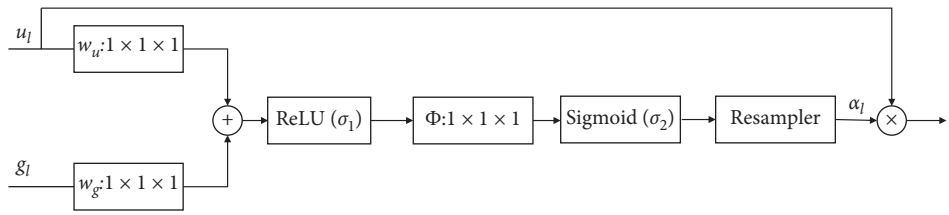


Figure 4: Schematic of the proposed additive AG.

Finally, the output of AGs is the multiplication of feature map and attention coefficient by elements, as shown in formula (5).

$$\hat{u}_l^{\text{up}} = u_l \times \alpha_l. \tag{5}$$

Attention coefficients tend to obtain large values in target organ regions and relatively small values in background regions, which can improve the accuracy of image segmentation.

3. Experimental Results

The experiments are performed on three datasets: DRIVE, ISIC 2018, and public dataset used in LUNA and the Kaggle Data Science Bowl 2017. The following performance indicators are adopted in this paper: True positive (TP), true negative (TN), false positive (FP), and false negative (FN), including accuracy (AC), sensitivity (SE), specificity (SP), and $F1$-score (F_1).

The accuracy rate is used to evaluate the accuracy of pixel classification and obtained by the following formula:

$$\text{AC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{6}$$

Sensitivity represents the proportion of samples that are predicted to be positive in the experimental results. It reflects the situation of positive samples. Sensitivity is calculated by the following formula:

$$\text{SE} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{7}$$

The specificity is calculated by the following formula:

$$\text{SP} = \frac{\text{TN}}{\text{TP} + \text{FP}}. \tag{8}$$

$F1$-score is used to measure the accuracy of binary classification model. It considers both the precision and recall of the classification model. It can be regarded as a harmonic average of model precision and recall. $F1$-score is calculated by the following formula:

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{9}$$

In addition, receiver operating characteristics (ROC) curve and precision recall (PR) curve are used to compare the performance of each network more intuitively. The values of area under curve (AUC) of both ROC curve and PR curve of each network are calculated in this paper.

3.1. Skin Lesion Segmentation. The ISIC dataset is published by the International Skin Imaging Collaboration (ISIC), which contains 2594 dermoscopy images of common skin pigmentation lesions. All the images have been annotated by the recognized skin cancer specialists. These annotations include dermoscopic features, which are used to identify the type of skin lesions of the known global and local morphological elements in the image. During the experiment, 1815 images are used for training, 259 images are used for verification, and 520 images are used as testing sets. The size of each image in ISIC is 700×900. Firstly, the input image is preprocessed into 256×256. Training images include original images and ground truth images labeled by professional physicians. Figure 5 shows the segmentation results of ISIC. The first column is the original images, the second column is the ground truth images, and the third column shows the segmentation result of R2AU-Net. The first line of the following skin lesion segmentation figures shows that R2AU-Net can accurately segment the dark skin lesions and will not be affected by the hair around the lesions. For the less obvious light-colored lesions in the second line, R2AU-Net can also segment the lesions well. It can be found that R2AU-Net can accurately segment the image of skin lesions, which is almost identical to the ground truth image. Table 1 shows the comparison of segmentation result among R2AU-Net and other improved versions of U-Net through $F1$-score, sensitivity, specificity, accuracy, and AUC value. R2AU-Net performs well in various indicators. For the dichotomy experiment, ROC curve and PR curve can intuitively compare the performance of each classifier. The AUC values of the ROC curve and PR curve are shown in Figures 6 and 7, respectively. The ROC curve tends to the upper left and the PR curve tends to the upper right, which shows the great performance of the segmentation model.

3.2. Retinal Vascular Segmentation. Images of DRIVE dataset are obtained from the diabetic retinopathy screening program in the Netherlands. Screening groups included 400 subjects aged between 18 and 24 who were diabetic. 40 color retina images are randomly selected. The doctor can diagnose, screen, treat, and evaluate a variety of cardiovascular and ophthalmic diseases, such as diabetes, hypertension, arteriosclerosis, and choroidal neovascularization, through the blood vessel segmentation from retinopathy images and the signs of retina blood vessel morphological properties. In the experiment, 20 samples are used for training and 20 samples are used for testing. The size of the original image is 565×584. Obviously, the number of samples is not enough to train a deep neural network model. Therefore, this paper randomly divided the input 20 training images into 190000

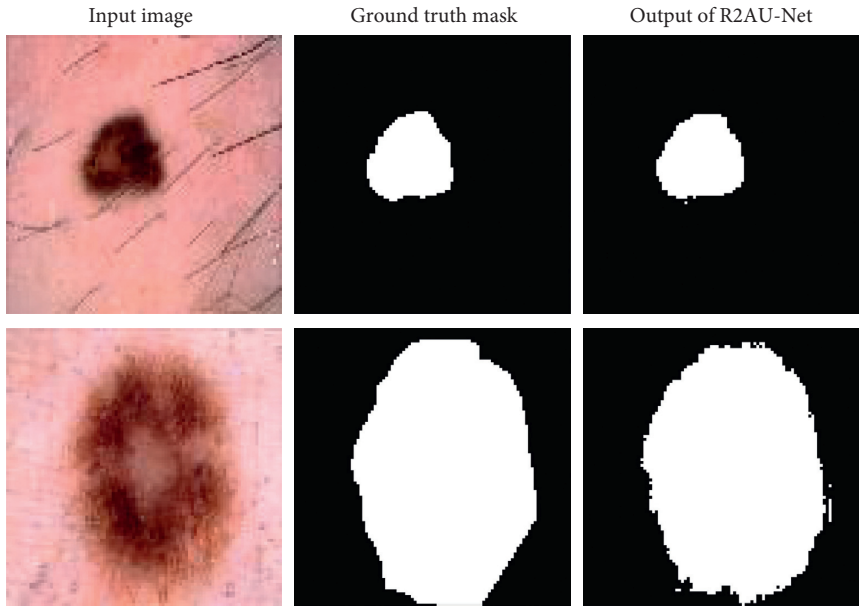


Figure 5: Segmentation results of skin lesions.

Table 1: Performance comparison of R2AU-Net and other networks on the ISIC dataset.

Methods	*F*1-score	Sensitivity	Specificity	Accuracy	AUC
SegNet [19]	0.7092	0.8056	0.8147	0.8121	0.8101
U-Net [3]	0.6954	0.7160	0.8636	0.8216	0.7898
Attention U-Net [20]	0.7541	0.7627	0.8966	0.8585	0.8297
RU-Net [21]	0.679	0.792	0.928	0.880	0.8374
R2U-Net [21]	0.691	0.726	0.971	0.904	0.8565
Attention Res U-Net	0.8545	0.8363	0.9519	0.9190	0.8941
R2AU-Net	0.8660	0.8214	0.9699	0.9277	0.8957

patches for training. Among them, 171000 patches are used for training set, and 19000 patches are used for testing set. The data size of the input network is 64×64. The segmentation result of the input image is shown in Figure 8. The first image is the original color image, the second image is the ground truth mask, and the third image is the segmentation result of the R2AU-Net output; most of the blood vessels at the end can still be segmented. Table 2 shows the results of comparative experiments on the DRIVE dataset, including *F*1-score, sensitivity, specificity, accuracy, and AUC value. From the experimental results, the performance of R2AU-Net is better than the traditional methods and the original U-Net. Figures 9 and 10 show the AUC value of both ROC curve and PR curve of each network.

3.3. Lung Segmentation. Public datasets used in LUNA and the Kaggle Data Science Bowl 2017 are provided by the National Cancer Research Center of the United States. This dataset consists of 2D and 3D images. The original size of lung CT images is 512×512, the number of which is 267. 134 images are used for training, 54 images are used for verification, and 79 images are used for testing set. Figure 11

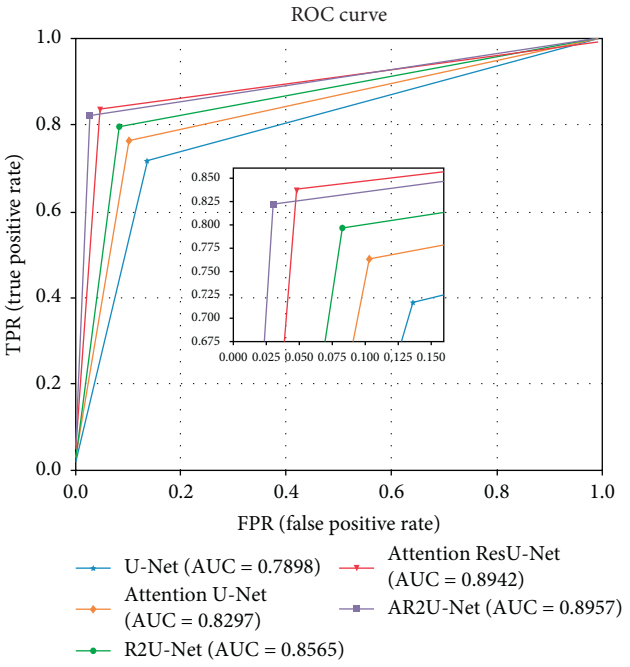


Figure 6: ROC curves of skin lesion segmentation of five models.

shows the segmentation results of R2AU-Net on the lung dataset. The first column is the input image, the second column is the ground truth mask, and the third column is the lung segmentation image of R2AU-Net. The third image of the first row shows that very small CT image of lung region is able to be segmented. The segmentation results of R2AU-Net are basically the same as the ground truth image. Table 3 shows the performance comparison of R2AU-Net and other improved versions of U-Net. Figures 12 and 13 show the AUC value of both ROC curve and PR curve of each network.

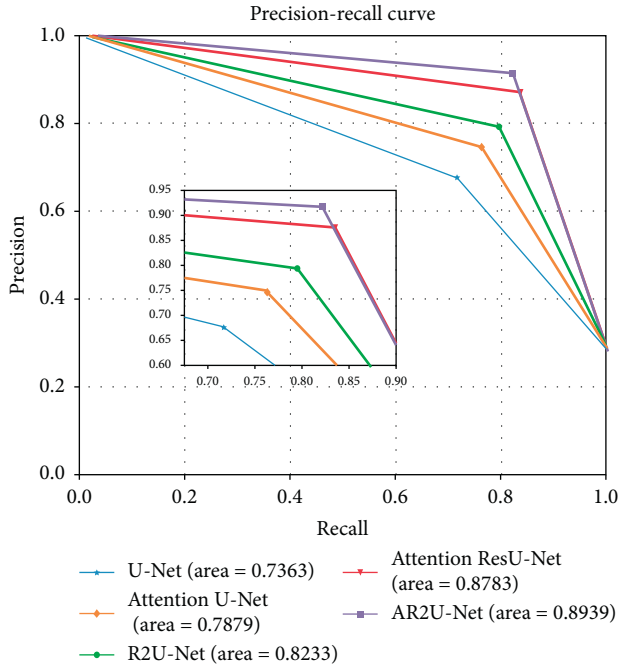


Figure 7: PR curves of skin lesion segmentation of five models.

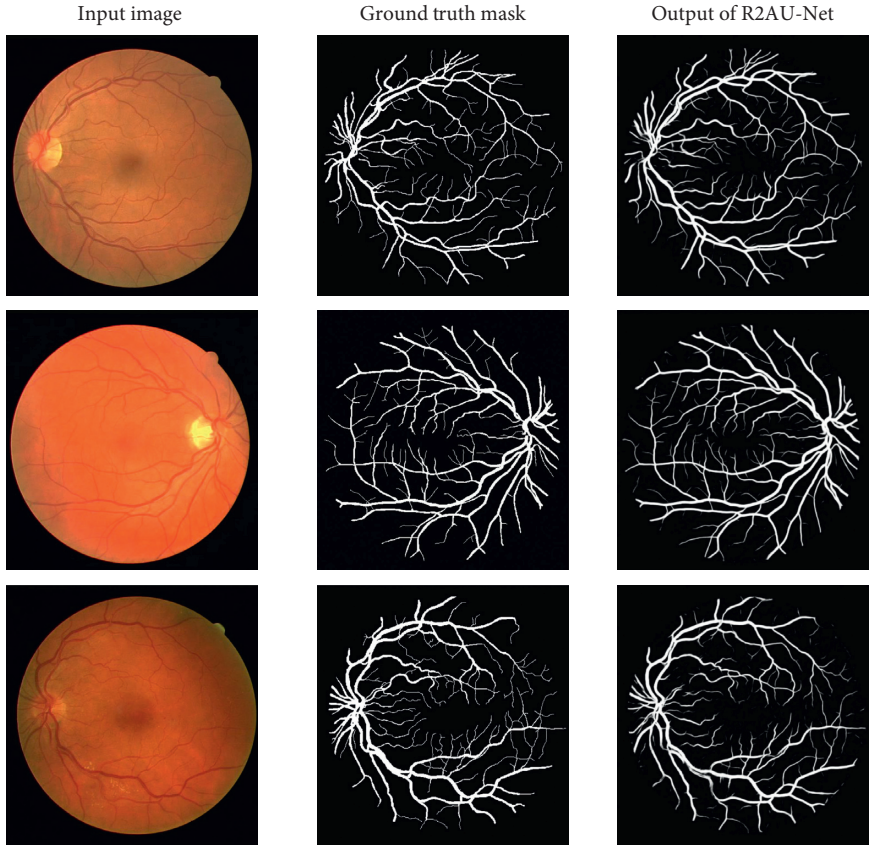


Figure 8: Retinal blood vessel segmentation results.

Table 2: Performance comparison of R2AU-Net and other networks on the DRIVE dataset.

Methods	*F*1-score	Sensitivity	Specificity	Accuracy	AUC
Hybrid features [22]	—	0.7252	0.9798	0.9474	0.9648
Trainable COSFIRE filters [23]	—	0.7655	0.9704	0.9442	0.9614
Three-stage filtering [24]	—	0.7250	0.9830	0.9520	0.9620
Deep model [25]	—	0.7763	0.9768	0.9495	0.9720
Cross-modality [26]	—	0.7569	0.9816	0.9527	0.9738
SegNet [19]	0.7992	0.7419	0.9833	0.9526	0.9752
U-Net [3]	0.8155	0.7908	0.9783	0.9544	0.9775
Attention U-Net [20]	0.8003	0.7272	0.9868	0.9538	0.9771
RU-Net [21]	0.8180	0.7999	0.9772	0.9547	0.9773
R2U-Net [21]	0.8187	0.7980	0.9779	0.9550	0.9775
R2AU-Net	0.8213	0.8036	0.9777	0.9555	0.9790

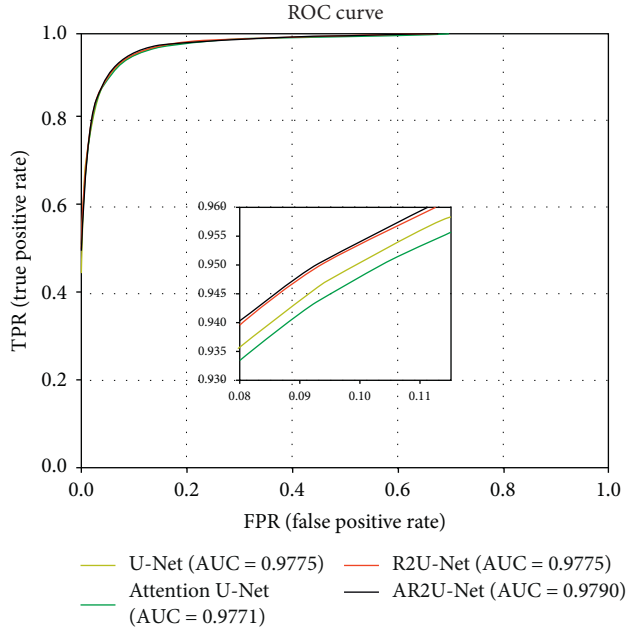


Figure 9: ROC curve of retinal blood vessel segmentation of four models.

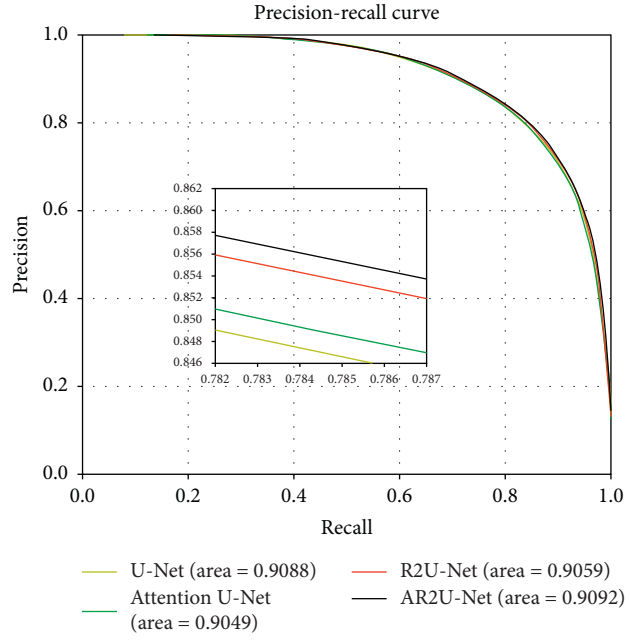


Figure 10: PR curve of retinal blood vessel segmentation of four models.

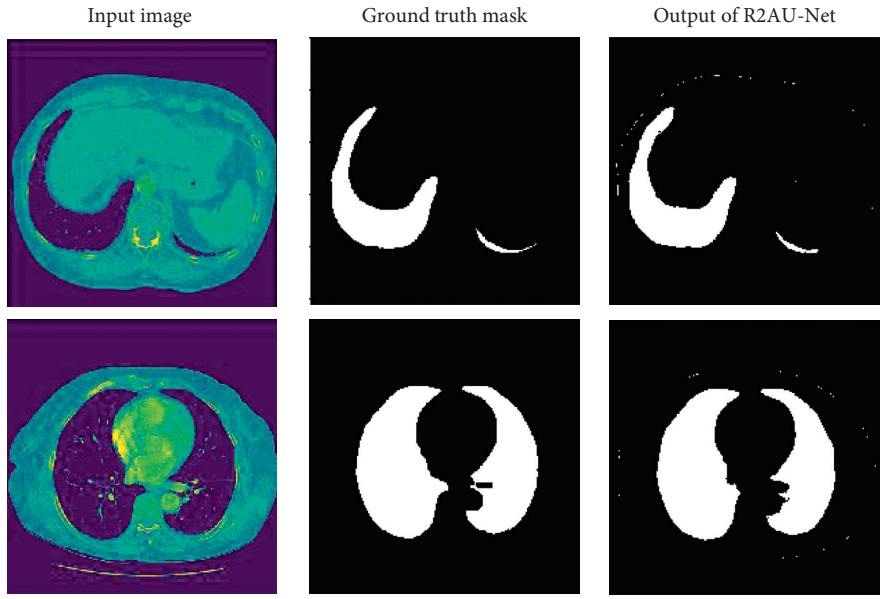


Figure 11: Results of the lung region segmented by R2AU-Net.

Table 3: Performance comparison of R2AU-Net and other networks on the lung dataset.

Methods	*F*1-score	Sensitivity	Specificity	Accuracy	AUC
SegNet [19]	0.9515	0.9240	0.9957	0.9820	0.9598
U-Net [3]	0.9714	0.9536	0.9977	0.9892	0.9756
Attention U-Net [20]	0.9780	0.9911	0.9915	0.9915	0.9914
RU-Net [21]	0.9638	0.9734	0.9866	0.9836	0.9800
R2U-Net [21]	0.9809	0.9842	0.9946	0.9926	0.9894
Attention Res U-Net	0.9811	0.9862	0.9942	0.9927	0.9902
R2AU-Net	0.9868	0.9874	0.9968	0.9950	0.9921

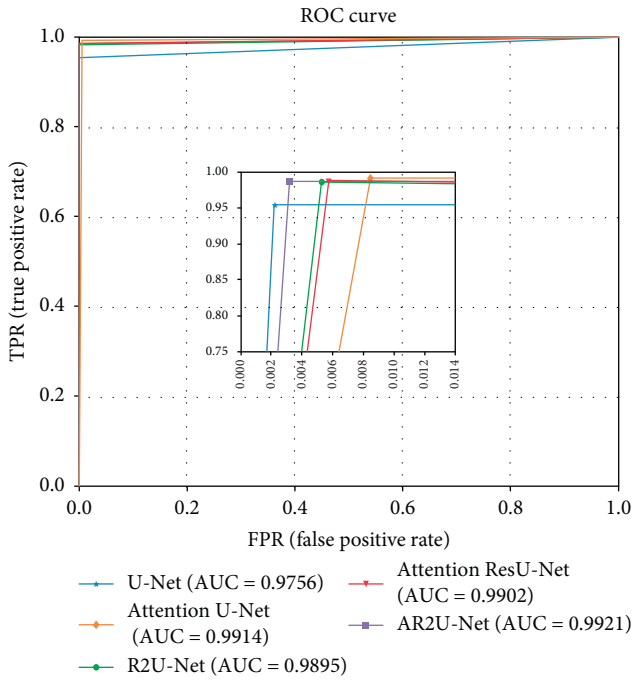


Figure 12: ROC curves of the lung region segmented by five models.

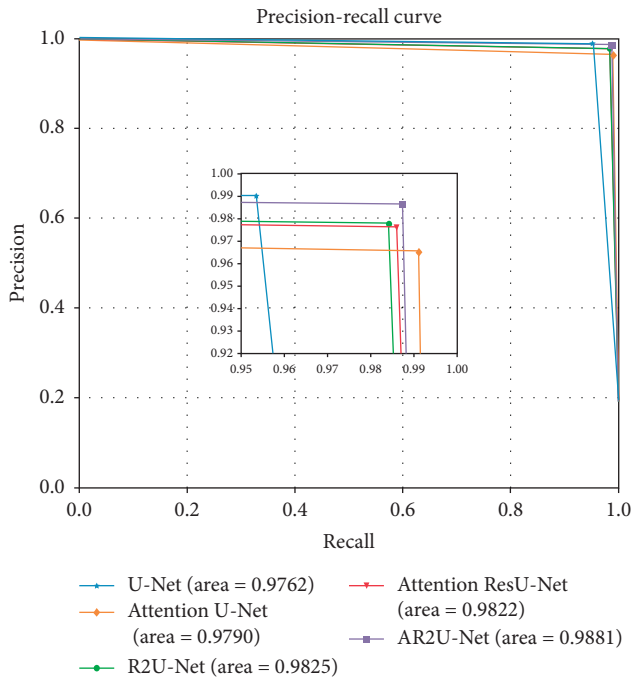


Figure 13: PR curves of the lung region segmented by five models.

4. Conclusion

In this paper, R2AU-Net is proposed for medical image segmentation. The recurrent residual convolutional block is used to enhance the ability of capturing context information, and AGs are added in the skip connections. Attention gates use deep features of decoding path as gating signal to modify shallow features and suppress feature response of background area, so that the network can obtain more accurate segmentation results. Moreover, BN is used to accelerate the convergence speed and stability of the network in the upsampling process. The experimental results of three datasets show that R2AU-Net has good performance in medical image segmentation.

Data Availability

The following are the links to the datasets used in this article: skin lesion segmentation (https://challenge2018.isic-archive.com/), retinal vascular segmentation (https://drive.google.com/file/d/17wVfELqgwbp4Q02GD247jJyjq6lwB0l6/view), and lung nodule segmentation (https://www.kaggle.com/kmader/finding-lungs-in-ct-data/data).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] D. C. Cirean, A. Giusti, L. M. Gambardella et al., "Deep neural networks segment neuronal membranes in electron microscopy images," *Advances in Neural Information Processing Systems*, vol. 25, pp. 2852–2860, 2012.

[2] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2015.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Aassisted Iintervention*, pp. 234–241, Munich, Germany, October 2015.

[4] M. Khened, V. A. Kollerathu, and G. Krishnamurthi, "Fully convolutional multi-scale residual DenseNets for cardiac segmentation and automated cardiac diagnosis using ensemble of Classifiers," *Medical Image Analysis*, vol. 51, pp. 21–45, 2019.

[5] C. Payer, D. Štern, H. Bischof, and M. Urschler, "Multi-label whole heart segmentation using cnns and anatomical label configurations," in *Statistical Atlases and Computational Models of the Heart. ACDC and MMWHS Challenges*, pp. 190–198, Springer, Berlin, Germany, 2018.

[6] H. R. Roth, H. Oda, Y. Hayashi et al., "Hierarchical 3D fully convolutional networks for multi-organ segmentation," 2017, https://arxiv.org/abs/1704.06382.

[7] F. Liao, M. Liang, Z. Li et al., "Evaluate the malignancy of pulmonary nodules using the 3D deep leaky noisy-or network," *IEEE Transactions on Neural Networks & Learning Systems*, vol. 30, no. 11, pp. 3484–3495, 2017.

[8] Z. Liu, Y.-Q. Song, V. S. Sheng et al., "Liver CT sequence segmentation based with improved U-net and graph Cut," *Expert Systems with Applications*, vol. 126, pp. 54–63, 2019.

[9] R. Azad, M. Asadi-Aghbolaghi, M. Fathy et al., "Bi-directional ConvLSTM U-Net with densley connected convolutions," in *Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition*, pp. 406–415, Seoul, Republic of Korea, October 2019.

[10] N. Abraham and N. M. Khan, "A novel focal tversky loss function with improved attention U-Net for lesion segmentation," in *Proceedings of the IEEE 16th International Symposium on Biomedical Imaging (ISBI)*, pp. 683–687, Venice, Italy, April 2019.

[11] K. He, X. Zhang, S. Ren et al., "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[12] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3367–3375, Boston, MA, USA, June 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *Computer Vision—ECCV 2016*, Springer International Publishing, Berlin, Germany, pp. 630–645, 2016.

[14] N. Ibtehaz and M. S. Rahman, "MultiResUNet: rethinking the U-Net architecture for multimodal biomedical image segmentation," *Neural Networks*, vol. 121, pp. 74–87, 2020.

[15] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, pp. 448–456, Lille, France, July 2015.

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, https://arxiv.org/abs/1409.0473.

[17] X. Wang, R. Girshick, A. Gupta et al., "Non-local neural networks," 2017, https://arxiv.org/abs/1711.07971.

[18] V. Mnih, N. Heess, A. Graves et al., "Recurrent models of visual attention," *Advances in Neural Information Processing Systems*, pp. 2204–2212, MIT Press, Cambridge, MA, USA, 2014.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: a deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[20] O. Oktay, J. Schlemper, L. L. Folgoc et al., "Attention U-Net: learning where to look for the pancreas," 2018, https://arxiv.org/abs/1804.03999.

[21] M. Z. Alom, M. Hasan, C. Yakopcic et al., "Recurrent residual convolutional neural network based on u-net (R2U-net) for medical image segmentation," 2018, https://arxiv.org/abs/1802.06955.

[22] E. Cheng, L. Du, Y. Wu, Y. J. Zhu, V. Megalooikonomou, and H. Ling, "Discriminative vessel segmentation in retinal images by fusing context-aware hybrid features," *Machine Vision and Applications*, vol. 25, no. 7, pp. 1779–1792, 2014.

[23] G. Azzopardi, N. Strisciuglio, M. Vento, and N. Petkov, "Trainable COSFIRE filters for vessel delineation with application to retinal images," *Medical Image Analysis*, vol. 19, no. 1, pp. 46–57, 2015.

[24] S. Roychowdhury, D. D. Koozekanani, and K. K. Parhi, "Blood vessel segmentation of fundus images by major vessel extraction and subimage classification," *IEEE Journal of*

Biomedical and Health Informatics, vol. 19, no. 3, pp. 1118–1128, 2015.

[25] P. Liskowski and K. Krawiec, "Segmenting retinal blood vessels With_newline deep neural networks," *IEEE Transactions on Medical Imaging*, vol. 35, no. 11, pp. 2369–2380, 2016.

[26] Q. Li, B. Feng, L. P. Xie et al., "A Cross-modality learning approach for vessel segmentation in retinal images," *IEEE Transactions on Medical Imaging*, vol. 35, no. 1, pp. 109–118, 2015.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6662337, 10 pages
https://doi.org/10.1155/2021/6662337

Research Article

Two-Level Multimodal Fusion for Sentiment Analysis in Public Security

Jianguo Sun,[1] **Hanqi Yin**,[1] **Ye Tian**,[1] **Junpeng Wu**,[1] **Linshan Shen**,[1] **and Lei Chen**[2]

[1]*College of Computer Science and Technology, Harbin Engineering University, Harbin, Heilongjiang 150001, China*
[2]*College of Engineering and Computing, Georgia Southern University, Statesboro, GA 30458, USA*

Correspondence should be addressed to Linshan Shen; shenlinshan@hrbeu.edu.cn

Received 28 December 2020; Accepted 21 May 2021; Published 4 June 2021

Academic Editor: David Megías

Large amounts of data are widely stored in cyberspace. Not only can they bring much convenience to people's lives and work, but they can also assist the work in the information security field, such as microexpression recognition and sentiment analysis in the criminal investigation. Thus, it is of great significance to recognize and analyze the sentiment information, which is usually described by different modalities. Due to the correlation among different modalities data, multimodal can provide more comprehensive and robust information than unimodal in data analysis tasks. The complementary information from different modalities can be obtained by multimodal fusion methods. These approaches can process multimodal data through fusion algorithms and ensure the accuracy of the information used for subsequent classification or prediction tasks. In this study, a two-level multimodal fusion (TlMF) method with both data-level and decision-level fusion is proposed to achieve the sentiment analysis task. In the data-level fusion stage, a tensor fusion network is utilized to obtain the text-audio and text-video embeddings by fusing the text with audio and video features, respectively. During the decision-level fusion stage, the soft fusion method is adopted to fuse the classification or prediction results of the upstream classifiers, so that the final classification or prediction results can be as accurate as possible. The proposed method is tested on the CMU-MOSI, CMU-MOSEI, and IEMOCAP datasets, and the empirical results and ablation studies confirm the effectiveness of TlMF in capturing useful information from all the test modalities.

1. Introduction

The main way of human communication includes language, visual, and acoustic, which can be generally rendered by speaking or writing, images or videos, and encoded phoneme information, respectively. Therefore, it is incomplete and limited to study the opinion, attitude, and sentiment of an entity only through the language or vision modal. Focusing on this point, multimodal research has been popularized and widely used in many fields of machine learning and artificial intelligence, such as tasks of sentiment analysis [1, 2], emotion recognition [3, 4], behavior recognition [5], video captioning [6], and medical diagnose [7–9]. Besides, in the field of public information security, microexpression recognition [10, 11] is the core process in the lie detection task. However, since the microexpressions are short duration and hard to capture by human and machine, most existing investigations are unsatisfactory. Accordingly, using the information complementarity strategy, i.e., applying the voices and audio modalities of the video as auxiliary information in a microexpression recognition task, can effectively increase the accuracy of lie detection.

Multimodal fusion is the concept that integrates information of multiple modalities by classification or prediction [12] and has become one of the most popular research interests in the field of multimodal machine learning. The main advantages of multimodal fusion can be summarized as follows. First, it can obtain more accurate predictions by observing the phenomenon from different modalities. Second, a multimodal system can catch the information that is invisible in individual modalities by multimodal fusion. Third, a multimodal system can still operate when one of the

modalities is missing. In previous research studies on multimodal fusion, the early, late, and hybrid fusion methods [13–15] have been widely employed to achieve fusion tasks. Take the early fusion method as an example, which can achieve the fusion objective by connecting the embeddings of text (T), audio (A), and video (V). However, this method cannot effectively extract the interactive features among the modalities.

On the other hand, the neural network has been extensively used for the multimodal fusion tasks due to its impressive advantages. First, neural network approaches are adapted in learning from a large amount of data. Second, the existing architectures of the neural network allow the end-to-end training that is based on the multimodal representation component and the fusion component. Moreover, the neural network can find the optimal solution quickly, learn complex decision boundaries, and has the ability of fault tolerance. By means of neural networks, Zadeh et al. [16] proposed the tensor fusion network (TFN), which learns both the intra-modality and intermodality dynamics end-to-end. Compared with the early fusion method used for the concatenation of multiple modalities embeddings, the TFN can effectively learn interactions between different features by outer product. Based on the TFN, low-rank multimodal fusion (LMF) [17], memory fusion network (MFN) [18], and multimodal transformer (MuLT) [19] have been proposed, which can further improve the processing efficiency and evaluation. In addition, it can be seen from these results that attaching both audio and video features to the same textual information can enable nontext information to be better understood, and in turn, the nontext information can impart greater meaning to the text [20]. Therefore, the multimodal fusion can be achieved according to correlations between those text-based features.

It should be mentioned that the long short-term memory (LSTM) network [21] was utilized in [18, 22] to process the sequence of data. The basic component of the architecture is a simple LSTM cell that contains three gates, namely, input gate, forgot gate, and output gate. The input gate determines the information to be remembered, and the forget gate selects the information to be dropped. The function of the output gate determines the output from the input listed in the memory. Based on these gates, the LSTM network can provide a prediction at each time unit, and it can learn long-term information more easily than the traditional recurrent neural network (RNN) by resorting to its self-loop mechanism. It should be pointed out that the LSTM network can only utilize the past input features of a specific period. For the future one, which plays a key role in the time-dependent sentiment analysis task, it is unavailable. For this reason, the BiLSTM network [23] has been popularized since it can learn the backward and forward long-term dependencies between small timestamps of the data sequence simultaneously. For example, in the field of natural language processing (NLP), the BiLSTM network was utilized for the context representation task [24] where more accurate evaluation can be obtained than the LSTM-based results [25]. In [26], the authors completed the acoustic modeling task utilizing the BiLSTM network and showed the state-of-the-art results on the TIMIT speech database.

Fusion in the decision level is a higher level of information fusion, which combines information from different data sources classified individually [27]. Generally, decision fusion for multisources data has the potential to improve classification accuracy, compared with the individual data sources. In previous research studies, decision fusion is widely used in several classification tasks [28]. For example, in the remote sensing image classification task, Mazher et al. [29] proposed a decision fusion method in land cover classification. The authors indicate that the proposed decision fusion method has desired generalization performance and can be also applied to other applications with different sensor data. In the medical diagnosis task, Agarwal and Bedi [30] proposed a fusion method using curvelet and wavelet transform and applied it to the medical diagnosis task that combines the images obtained by computed tomography (CT) scan and magnetic resonance imaging (MRI). The experiment shows that the results of two-level fusion are better than those of one feature-level fusion and decision-level fusion. In the natural resource prediction task, DSS was applied to the problem of sustainable greenhouse management [31] and regarded as an effective strategy to balance the sustainability and the profitability of productions.

Motivated by the above discussion, this study proposes a new modal fusion method named TlMF, which produces unimodal embeddings by using a CNN-BiLSTM neural network and achieves the information fusion of text and audio/video embedding based on tensor fusion and decision fusion stages. The cores of our method include (1) a tensor fusion network used to fuse text data with video and audio data, respectively, and (2) a decision-level fusion strategy, which can fuse the classification results. Then, three datasets of multimodal sentiment analysis and emotion classification, i.e., CMU-MOSI [32], CMU-MOSEI [33], and IEMOCAP [34] are used for experiments to evaluate the effectiveness of the proposed methods.

The rest of this study is organized as follows: Section 2 introduces the proposed model; the experimental setup and the experimental results are given in Section 3, and the conclusion is provided in Section 4.

2. Proposed Method

Figure 1 shows the main diagram of the TlMF model that is established on the stages of data preparation, feature extraction, tensor fusion, and decision fusion. To make this framework clearer, detailed descriptions of some components are given in this section.

2.1. CNN-BiLSTM. A convolution neural network (CNN) is utilized to learn features from each modality after the data preparation stage. In this study, the input sequences of the text, audio, and video feature embeddings are defined as $H_t \in R^{d_t \times l_t}$, $H_a \in R^{d_a \times l_a}$, and $H_v \in R^{d_v \times l_v}$, respectively. To extract the features from these input sequences, a 1D temporary convolution layer is used for the time dimension of each input vector. And the outputs of this convolution layer are denoted as $H_{t1} \in R^{d_{t1} \times l_t}$, $H_{a1} \in R^{d_{a1} \times l_a}$, and

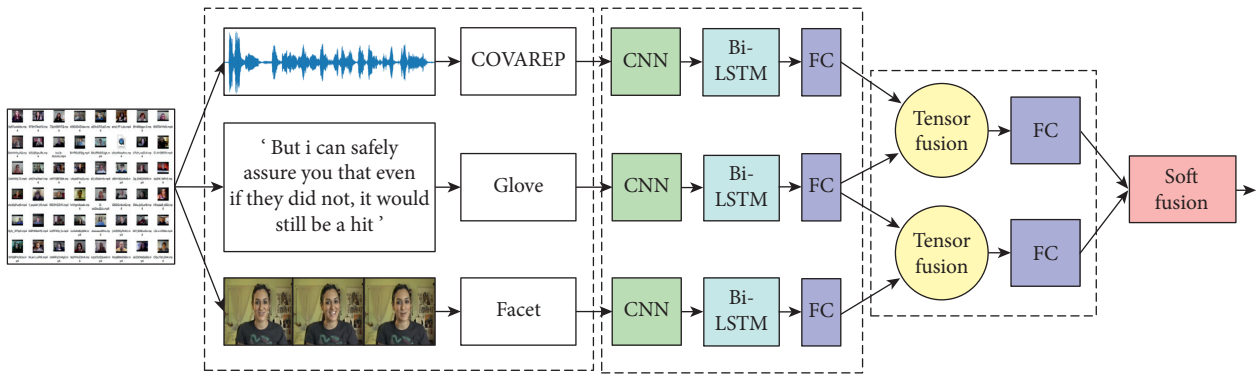


Figure 1: Overall architecture of TlMF. The first stage is data preparation, which turns the raw data into a unimodal sequence for text, audio, and video modalities. Once the unimodal sequence is obtained, the unimodal features can be learned by the second stage, which can extract features from each modality. Then, the tensor fusion layer is used to fuse the text-based audio feature H_{ta} and the text-based video feature H_{tv}. Finally, a decision fusion layer is employed to improve the accuracy of classification and prediction in the sentiment analysis task.

$H_{v1} \in R^{d_{v1} \times l_v}$, respectively. For ease of calculation, we assume $d_{t1} = d_{a1} = d_{v1}$.

The output of the CNN is then processed by the BiLSTM layer, which is extended from LSTM and can learn the context correlation from time-series data. It should be pointed out that LSTM has two special structures: cell state and gate. The latter consists of the input gate, the forget gate, and the output gate, which control the update, maintenance, and deletion of cell state, respectively. By denoting C_t as the current cell state of LSTM and h_t as the corresponding hidden state at the instant t, the forward propagation process of LSTM can be summarized as follows.

The first step is to update the output f_t of the forget gate, which controls whether to forget the cell state of the upper layer with a certain probability. Note that the input of the forget gate includes the hidden state h_{t-1} of the previous sequence and the current sequence data $x_t \in \{H_{t1}, H_{a1}, H_{v1}\}$. Thus, the rule of the output is described by the following equation:

$$f_t = \sigma\left(\theta_f \left[h_{t-1}, x_t\right] + b_f\right), \tag{1}$$

where θ_f and b_f are the sets of weight matrices and biases vectors in the forget gate, respectively; σ represents the sigmoid activation function.

The second step is to update the information stored in the current cell state C_t. For this purpose, two parts need to be considered. First, by sigmoid activation function, the output i_t of the input gate is generated, which contains the values to be updated. Then, a tanh layer creates a vector of new candidate values a_t that could be added to the state, as shown in the following equations:

$$i_t = \sigma\left(\theta_i \left[h_{t-1}, x_t\right] + b_i\right), \tag{2}$$

$$a_t = \tanh\left(\theta_a \left[h_{t-1}, x_t\right] + b_a\right), \tag{3}$$

where θ_i and b_i mean the sets of weight matrices and biases vectors in the input gate, respectively. Note that once f_t, a_t, and i_t are determined, the current cell state C_t can be updated from the last cell state C_{t-1} according to

$$C_t = \left(C_{t-1} \otimes f_t \oplus i_t \otimes a_t\right), \tag{4}$$

where $\otimes$ and $\oplus$ mean the Hadamard product and the concatenation operator, respectively.

Now, we can determine the desired output based on the cell state C_t. Let θ_o and b_o be the sets of weight matrices and biases vectors in the output gate, respectively. The main procedure is composed of two steps shown in equations (5) and (6). First, the output gate is to select the parts to be output from C_t, and the corresponding output is given by o_t. Then, by using the Hadamard product between $\tanh(C_t)$ and o_t, the desired output can be obtained.

$$o_t = \sigma\left(\theta_o \left[h_{t-1}, x_t\right] + b_o\right), \tag{5}$$

$$h_t = o_t \otimes \tanh\left(C_t\right). \tag{6}$$

Finally, the prediction output of the current sequence is updated in accordance with

$$y_n = \varphi\left(\theta_y h_t + b_y\right), \tag{7}$$

where θ_y and b_y represent the sets of weight matrices and biases vectors in the predict layer, respectively; φ is the output activation function, e.g., Softmax in classification task or tanh in prediction task.

So far, the forward propagation of LSTM has been described in detail. It can be seen from equations (1)–(7) that the LSTM network can only utilize the previous input features of a specific period. This characteristic of LSTM greatly restricts its application in the sentiment analysis task. In order to simultaneously learn the previous and future context from sequence data, BiLSTM is considered, which combines two separate hidden LSTM layers with the opposite directions of the same input.

The architecture of the BiLSTM network is shown in Figure 2, where $\overrightarrow{h}_t = (\overrightarrow{h}_{t_0}, \overrightarrow{h}_{t_1}, \ldots, \overrightarrow{h}_{t_n})$, $\overleftarrow{h}_t = (\overleftarrow{h}_{t_0}, \overleftarrow{h}_{t_1}, \ldots, \overleftarrow{h}_{t_n})$, and $x_t = (x_{t_0}, x_{t_1}, \ldots, x_{t_n})$. The main procedure of BiLSTM can be described by

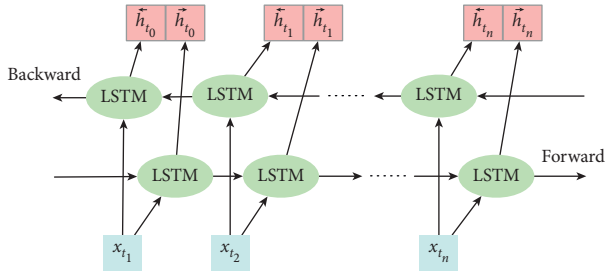


Figure 2: The architecture of the BiLSTM network.

$$\begin{aligned}
\overrightarrow{h}_t &= \sigma\left(\theta_{\overrightarrow{h}}\left[\overrightarrow{h}_{t-1}, x_t\right] + b_{\overrightarrow{h}}\right),\\
\overleftarrow{h}_t &= \sigma\left(\theta_{\overleftarrow{h}}\left[\overleftarrow{h}_{t-1}, x_t\right] + b_{\overleftarrow{h}}\right),\\
\left(\overrightarrow{h}_0, \overleftarrow{h}_0\right), \ldots, \left(\overrightarrow{h}_t, \overleftarrow{h}_t\right) &= \mathrm{BiLSTM}\left(x_0, x_1, \ldots, x_t\right),\\
y_t &= \sigma\left(\theta_{y_t \overrightarrow{h}}\overrightarrow{h}_t + \theta_{y_t \overleftarrow{h}}\overleftarrow{h}_t + b_{y_t}\right),
\end{aligned} \tag{8}$$

where $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ are the forward hidden sequence and the backward hidden sequence, respectively, and can be obtained from equations (5)–(7). $\theta_{\overrightarrow{h}}$, $\theta_{\overleftarrow{h}}$, $\theta_{y_t \overrightarrow{h}}\overrightarrow{h}_t$, and $\theta_{y_t \overleftarrow{h}}\overleftarrow{h}_t$ are the sets of weight matrices; $b_{\overrightarrow{h}}$, $b_{\overleftarrow{h}}$, and b_{y_t} represent the sets of biases vectors. The output vector $y_t = (y_0, y_1, \ldots, y_t)$ of the hidden layer is the concatenation of $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ and satisfies $y_t = [\overrightarrow{h}_t, \overleftarrow{h}_t]$.

In this study, each modal uses a BiLSTM block, which consists of three-stacked BiLSTM layers, two fully connection layers, a Maxpool layer, and a tanh layer. The output y_t from the upper layer becomes the input of the lower layer. Since the time series is limited, the computation load of the three BiLSTM layers will be not increased.

In the CNN-BiLSTM feature extraction network, the preprocessing is performed through a CNN subnetwork to learn local features, such that a shorter series with high-level features can be obtained. Then, a separate BiLSTM subnetwork is trained for different modalities. In this case, the output tensors of text, audio, and video are defined as $H_{t2} \in R^{d_{t2}}$, $H_{a2} \in R^{d_{a2}}$, and $H_{v2} \in R^{d_{v2}}$, respectively.

2.2. Tensor Fusion. In this study, the TFN [16] is adopted, which can effectively learn the interactions between different features by outer product. Moreover, the employment of outer product can lead to none learnable parameters and a low possibility of overfitting despite the high-dimensional output tensor. Therefore, we choose the outer product to fusion our data sequence.

It should be mentioned that since text modal contains more considerable sentiment-related information than video and audio modalities. For this reason, we fuse the text information with audio and video separately and denote text-audio and text-video features as H_{ta} and H_{tv} respectively. Specifically, after the CNN-BiLSTM network, we use the tensor fusion layer to learn interactions between different modalities. The inputs of tensor fusion layer are the outputs from the CNN-BiLSTM network. The calculation can be performed by the following equations:

$$H_{\mathrm{ta}} = \left[H_{t2} \otimes H_{a2}\right], \tag{9}$$

$$H_{\mathrm{tv}} = \left[H_{t2} \otimes H_{v2}\right]. \tag{10}$$

This process can be described by Figure 3. The fusion information $H_{\mathrm{ta}} \in R^{d_{t2} \times d_{a2}}$ and $H_{\mathrm{tv}} \in R^{d_{t2} \times d_{v2}}$ can be obtained after the tensor fusion layer. According to [16], since the output neurons of tensor fusion are easy to interpret and very meaningful in semantics, it is easy for the subsequent layers of the network to decode the meaningful information.

2.3. Decision Fusion. Fusion at the decision level between the classification results of the text-audio and text-video is considered to further improve the results. According to posterior probabilities, each classifier can yield a scoring matrix as the output of the Softmax layer to characterize the confidence level that the network chooses a specific class as the correct output class. Soft fusion is a method to get the new prediction label by fusing the scoring matrix from classifiers. Since soft fusion can increase accuracy in the minimum and worst case, it has been widely used in the medical diagnose field [9, 35]. In this study, the soft fusion method, which can incorporate the scores from separate fusion modalities, is applied to generate the new prediction label in the sentiment analysis task. The weighted combination of the two fusion modalities scores is shown in the following equation:

$$S_F(c) = W_{\mathrm{ta}} \cdot S_{\mathrm{ta}}(c) + W_{\mathrm{tv}} \cdot S_{\mathrm{tv}}(c), \tag{11}$$

where W_{ta} and W_{tv} denote the weight of text-audio modal and text-video modal, respectively, and satisfy $W_{\mathrm{ta}} = W_{\mathrm{tv}}$ at the initial time. $S_{\mathrm{ta}}(c)$ is the score matrix of the text-audio modal for the prediction of class c, $S_{\mathrm{tv}}(c)$ represents the score matrix of the text-video modal, and $S_F(c)$ stands for the final classification results.

3. Experiment and Discussion

3.1. Datasets. The proposed method is tested by using three public multimodal datasets: CMU Multimodal Corpus of Sentiment Intensity (CMU-MOSI) [32] and CMU Multimodal Opinion Sentiment and Emotion Intensity (CMU-MOSEI) [33] datasets for sentiment analysis task and Interactive Emotional Dyadic Motion Capture (IEMOCAP) [34] dataset for emotion classification task. The detailed descriptions of these datasets are given as follows.

CMU-MOSI is built on 93 YouTube movie review videos that are segmented into 3,702 utterance segments including 2,199 opinion video segments. Each segment is annotated by a scale in (−3, 3) to reflect sentiment intensity, where −3 and 3 represent the extremely negative and extremely positive sentiments, respectively. These segments are rigorously annotated

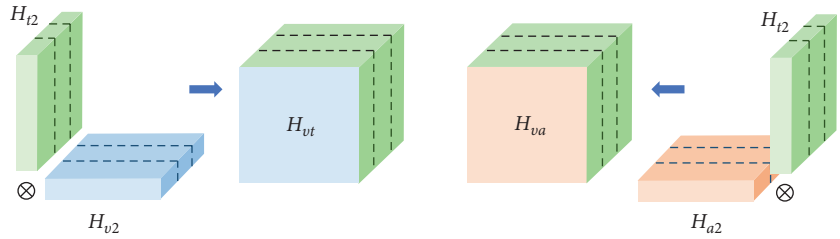


Figure 3: Tensor fusion.

according to the subjective sentiment intensity of the visual features and audio features. Moreover, these segments divided into three parts such as train, validation, and test are 16326, 1871, and 4659 data samples, respectively. The raw data and the distribution of sentiment intensity are shown in Figures 4 and 5, respectively. From Figure 5, we can clearly figure out that the number of segments in different sentiment intensity is relatively average.

CMU-MOSEI is the largest dataset of multimodal sentiment analysis tasks. It consists of 23453 sentence utterance video segments from more than 1000 online YouTube speakers and 250 topics. Each segment video is transcribed and properly punctuated, which can be treated as an individual multimodal example. In this case, train, validation, and test partitions contain 16326, 1871, and 4659 data samples, respectively. Figure 6 shows the part of raw data, and the sentiment intensity distributed in (−3, 3) is given in Figure 7, where −3 and 3 represent the extremely negative and extremely positive sentiments, respectively. It is clear from Figure 7 that most segments are labeled with (0, 1), which implies the sentiment intensity is weakly positive. The uneven distribution of the sentiment intensity corresponds to people's commenting habits.

IEMOCAP is an acted, multimodal, and multispeaker database, which has a free academic license, long duration, and good emotion label. It contains about 12 hours of 302 videos, in which speakers performed 9 different emotions: angry, excited, fear, sad, surprised, frustrated, happy, disappointed, and neutral. In this study, happy, sad, angry, and neutral emotions are chosen for experiment, and the distribution is shown in Figure 8. These considered videos are divided into 4444 segments with emotion annotations. In this case, train, validation, and test partitions contain 2717, 789, and 938 data samples, respectively.

3.2. Multimodal Features. For the above datasets, the unimodal features are extracted from the text, audio, and video data modalities by using global vectors for word representation (Glove) [36], COVAREP [37], and Facet, respectively. Moreover, these unimodal features can also be provided by the CMU-MultimodalSDK.

Text features is extracted by Glove, which is an unsupervised learning algorithm for obtaining vector representations of words. The dimension of each text embedding extracted by Glove is 300 for different inputs in the above datasets.

Audio features are extracted by utilizing COVAREP, which is a collaborative and freely available repository of speech processing algorithms. By using COVAREP, we can obtain low-level acoustic features including 12 Mel-frequency cepstral coefficients, pitch tracking, voiced/unvoiced segmenting features, glottal source parameters, and peak slope parameters. Each audio feature is extracted on 25-ms frames with a 5-ms shift, and its dimension is 74 for each dataset.

Video features consist of 35 facial action units extracted from each frame by using Facet, which is widely used for extracting facial expression features such as basic and advanced emotions. Thus, the dimension of each video frame feature is 35 for each dataset.

3.3. Performance Metrics. In this section, to show the advantages of the proposed method over the existing methods [16, 17, 19] in multiclass classification and prediction tasks, the following performance metrics are chosen for different datasets.

The CMU-MOSI and CMU-MOSEI datasets are labeled in the range of (−3, 3). According to the suggestion of the authors of the datasets, these labels can be divided into different groups. In this paper, two groups are considered. The labels in the first group are divided into two categories: the negative sentiment and the positive sentiment, which are with the labels in the range of (−3, 0) and (0, 3), respectively. In this case, we choose the binary accuracy (Acc-2) and F-score as the performance metrics on the CMU-MOSI and CMU-MOSEI datasets. The 7-class accuracy (Acc-7) is used as the performance metrics in the other group which have 7 sentiment score classification in (−3, 3). In addition, the mean absolute error (MAE) and the correlation between the prediction results and the ground truth label are chosen in these two datasets.

The IEMOCAP dataset are labeled in 9 different emotions, where happy, sad, angry, and neutral emotions are chosen in this study. For this emotion classification task, F-score and the binary accuracy (Acc-2) are selected as the performance metrics in each emotion.

3.4. Training Setup. The proposed method is implemented by open-source PyTorch framework, and it is tested and evaluated on the computer with Intel (R) Xeon (R) Silver 4214 at 2.20 GHz CPU, TITAN RTX GPU with 24 GB memory, and 64 GB computer memory. In addition, the hyperparameter is configured for different datasets, where the learning rate and the batch size vary in the range of

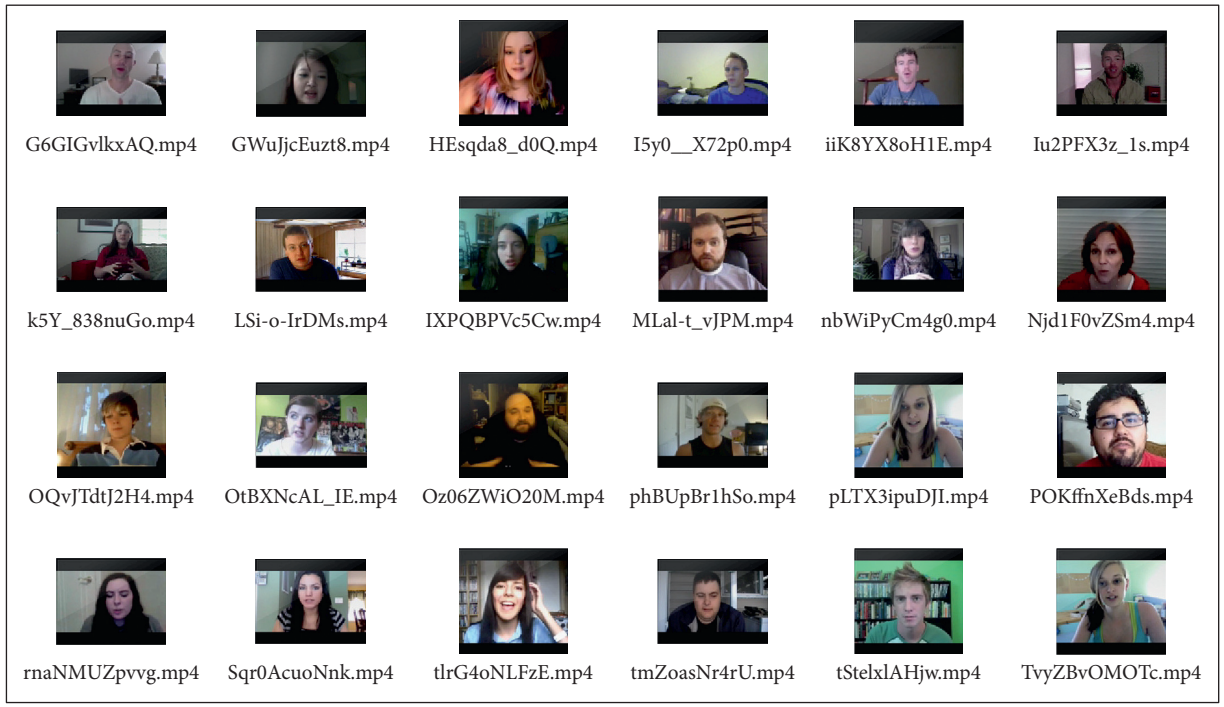


Figure 4: Part of raw videos in the CMU-MOSI dataset.

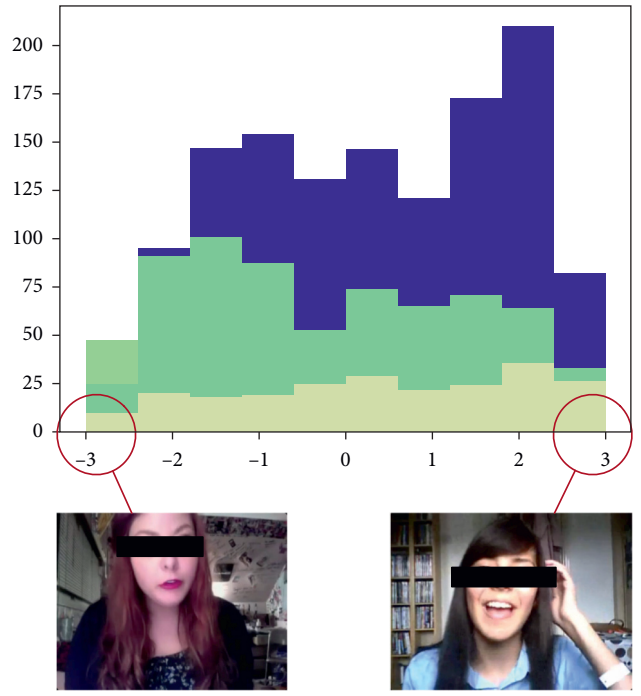


Figure 5: The sentiment intensity distribution of the CMU-MOSI dataset. Blue, yellow, and green represent train, valid, and test data. −3 and 3 represent the extremely negative and extremely positive sentiments, respectively.

$(10^{-3}, 10^{-5})$ and $(16, 128)$, respectively, and the epochs are trained 40–100 times by the model. ReLU and sigmoid are used as the activation functions. Mean square error is used as the loss function, and Adam is used as the optimizer. After optimizing the loss function, we use back propagation to update the parameters in the whole network.

3.5. Comparison between TlMF and the Existing Methods. In this subsection, the comparison between the proposed method and the existing methods, i.e., the tensor fusion network (TFN) [16], low-rank multimodal fusion (LMF) [17], and multimodal transformer (MuLT) [19], is made based on the experimental results of the sentiment analysis

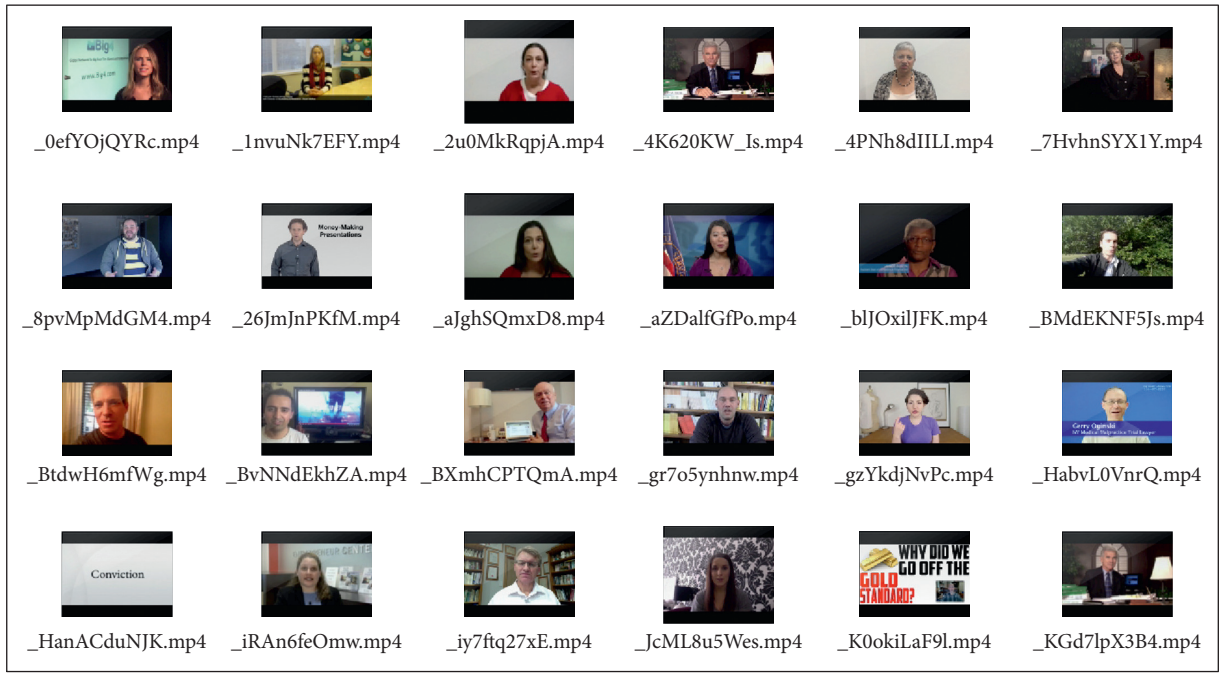


Figure 6: Part of raw videos in the CMU-MOSEI dataset.

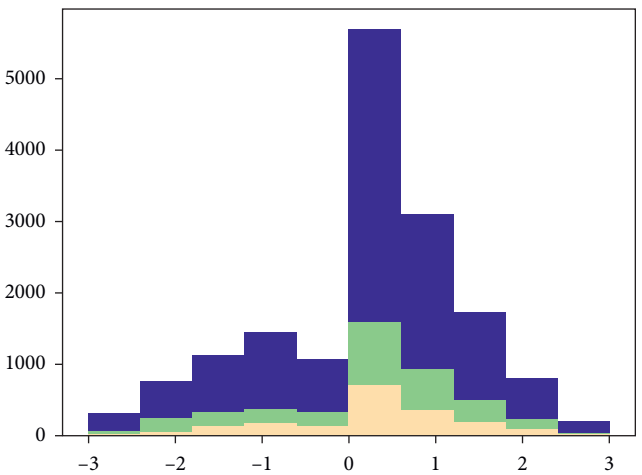


Figure 7: The sentiment intensity distribution of the CMU-MOSEI dataset. Blue, yellow, and green represent train, valid, and test data, respectively.

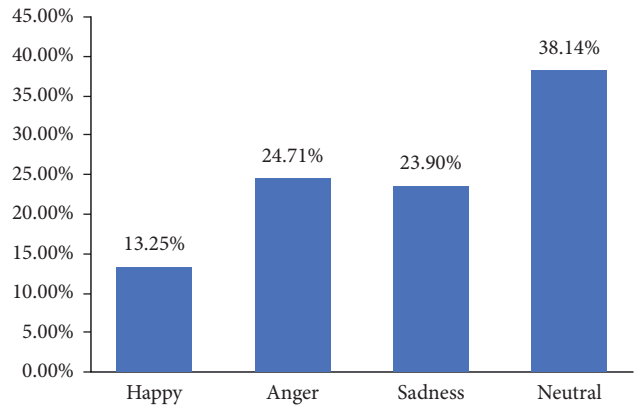


Figure 8: Emotion distribution of the IEMOCAP dataset.

task. It is known from [16] that the TFN combines individual modal's embeddings via calculating three different outer product subtensors such as unimodal, bimodal, and trimodal, and all subtensors are to be flattened as multimodal embedding vectors. As for LMF, it learns the multimodal embedding based on the similar tensor processing to that of the TFN, but with an additional low-rank factor for reducing computation memory. Different from the TFN and LMF, MuLT focus on interactions between multimodal sequences and latently adapt streams from one modality to another and uses the currently popular transformer model to transform one modal into another for modal fusion. In [19], MuLT has achieved state-of-art results in recent years on the multimodal fusion field. Thus, the comparison with MuLT can further test the superiority of TlMF on the sentiment analysis and the emotion classification. To guarantee the comparison fair, the three methods are tested under the same features and experimental environment.

In teams of the sentiment analysis task, the experiments are performed based on the CMU-MOSI and CMU-MOSEI datasets, and the corresponding comparison results are given in Table 1, respectively. From Table 1, it is seen that TlMF can obtain higher Acc-2, F-score, Acc-7, and Corr and lower MAE than TNF, LMF, and MuLT in the CMU-MOSI dataset. In the CMU-MOSEI dataset, it is also seen that TlMF can obtain higher Acc-2, F-score, Acc-7, and Corr than TNF, LMF, and MuLT. In other words, the proposed method is more effective in achieving accurate prediction than the existing methods under these two datasets. On the other hand, the comparison experimental results on the IEMOCAP datasets are given in Table 2. By observing this table, it is clear that the proposed method leads to higher

Table 1: Comparison experimental results for sentiment analysis on the CMU-MOSI and CMU-MOSEI datasets.

Metric	Acc-2	F-score	MAE	Acc-7	Corr
Case of the CMU-MOSI dataset					
TFN	80.82	80.77	0.901	23.94	0.698
LMF	82.53	82.47	0.917	33.23	0.695
MuLT	79.11	79.10	0.977	33.23	0.666
TlMF	92.28	92.26	0.373	65.49	0.933
Case of the CMU-MOSI dataset					
TFN	78.57	78.09	0.623	47.21	0.648
LMF	78.03	78.18	0.609	48.02	0.657
MuLT	79.35	79.29	0.631	48.45	0.647
TlMF	79.46	79.46	0.645	48.88	0.669

Table 2: Comparison experimental results for emotion classification on IEMOCAP.

Emotions	Happy		Sad		Angry		Neutral	
Metric	ACC-2	F-score	ACC-2	F-score	ACC-2	F-score	ACC-2	F-score
TFN	82.66	82.03	80.63	80.75	82.11	82.03	65.03	65.90
LMF	82.62	83.38	79.53	80.15	82.62	83.38	63.48	67.05
MuLT	85.60	78.96	79.42	70.31	75.79	65.36	59.59	50.33
TlMF	97.86	97.81	96.98	96.96	97.82	97.81	96.68	96.65

Acc-2 and F-score than the existing method in different emotions, which implies that more accurate classification can be achieved by TlMF. From the above discussion, it can be concluded that the proposed method has great superiority on accuracy of prediction and classification over the TFN, LMF methods when it is used for sentiment analysis.

3.6. Ablation Study. In this subsection, two series of ablation studies that compare different TlMF's variants are performed on the basis of the CMU-MOSI, CMU-MOSEI, and IEMOCAP datasets. The first series is to verify the effectiveness of the data fusion strategy in tensor fusion. For this purpose, we set text, audio, and video as the method's input separately and neglect the tensor fusion and decision fusion, such that the outputs corresponding to different inputs can be directly obtained from the downstream classifier. In this case, the text-only, audio-only, and video-only variants are denoted by TlMF_t, TlMF_a, and TlMF_v, respectively. In the second series of ablation studies, the influence of the tensor fusion and the decision fusion on the performance of the proposed method is illustrated by the following two cases:

Case 1: set text, audio, and video as the input. Discard the tensor fusion from TlMF and denote this variant as TlMF_{ntf}. The effectiveness of the tensor fusion can be confirmed by comparing the performance metrics of TlMF_{ntf} and TlMF.

Case 2: set the input as the same as in Case 1 and discard the decision fusion from TlMF. To keep the dimension of the output unchanged, the average-fusion is adopted to fuse the classification results of the upstream classifiers. In this case, the variant is defined as TlMF_{avg}, and the effectiveness of the decision fusion can be verified by comparing the performance metrics of TlMF_{avg} and TlMF.

Based on the above discussion, the experimental results of the ablation studies are provided in Tables 3 and 4. Specifically, Table 3 depicts the comparison results between TlMF and its variants in the sentiment analysis task when different datasets are utilized. It can be seen from the first three rows of these two subtables that Acc-2 and F-score of the text-only method are larger than those of the audio-only and video-only methods, which implies that the text modal can achieve more accurate sentiment analysis than the audio modal and video modal. The main contribution to this phenomenon is that the intrinsic structure of text is more adapted to emotional expression. Thus, it can represent language information better than the other modalities in the sentiment analysis task. On the other hand, by comparing the results of TlMF_{ntf} and TlMF (Case 1), it is clear that with the utilization of the tensor fusion, Acc-2, F-score, Acc-7, and MAE are, respectively, improved by 2.6%, 0.44%, 4.2%, and 4.2% on the CMU-MOSI dataset and by 3.96%, 4.1%, 5.01%, and 8.8% on the CMU-MOSEI dataset, which fully demonstrates the advantage of the tensor fusion in improving prediction accuracy of the sentiment analysis. Moreover, following a similar line to the analysis of Case 1, the effectiveness of the decision fusion can be confirmed according to the comparison between TlMF_{avg} and TlMF (Case 2).

For the emotion classification task, Table 4 provides the comparison results between TlMF and its variants on different emotions. By observing this table, there are two points worth emphasizing. First, it is seen from the first three rows that the text-only method leads to higher Acc-2 and F-score than the other methods for different emotions, that is, the text modal can achieve more accurate emotion classification than the audio modal and video modal. Second, by comparing the performance metrics of TlMF_{ntf} in Case 1 and TlMF_{avg} in Case 2 with those of TlMF, the effectiveness of the tensor fusion and the decision fusion can be clearly verified,

Table 3: Ablation experimental results for sentiment analysis on the CMU-MOSI and CMU-MOSEI datasets.

Metric	Acc-2	F-score	MAE	Acc-7	Corr
Case of the CMU-MOSI dataset					
TlMF_t	89.43	89.41	0.534	56.15	0.869
TlMF_a	77.33	77.31	0.983	30.37	0.51
TlMF_v	79.52	79.86	0.810	41.35	0.683
TlMF_{ntf}	90.08	90.06	0.512	56.15	0.886
TlMF_{avg}	89.60	89.62	0.553	52.95	0.886
TlMF	92.28	92.26	0.373	65.49	0.933
Case of the CMU-MOSI dataset					
TlMF_t	78.59	78.74	0.647	47.95	0.611
TlMF_a	62.27	64.52	0.717	41.36	0.593
TlMF_v	64.85	68.59	0.706	42.58	0.537
TlMF_{tnf}	75.53	76.86	0.716	44.53	0.534
TlMF_{avg}	77.35	77.97	0.665	46.98	0.581
TlMF	79.46	79.46	0.645	48.88	0.669

Table 4: Ablation experimental results for emotion classification on the IEMOCAP dataset.

Emotions	Happy		Sad		Angry		Neutral	
Metric	Acc-2	F-score	Acc-2	F-score	Acc-2	F-score	Acc-2	F-score
TlMF_t	97.46	97.41	95.14	95.08	94.58	94.54	93.48	93.42
TlMF_a	87.55	81.75	74.60	63.75	77.95	75.26	67.90	62.87
TlMF_v	94.11	93.40	92.08	91.62	91.97	91.58	86.52	85.84
TlMF_{ntf}	96.61	96.57	87.67	87.55	90.79	90.76	91.24	90.97
TlMF_{avg}	97.16	97.03	95.73	95.68	97.05	97.03	95.98	95.97
TlMF	97.86	97.81	96.98	96.96	97.82	97.81	96.68	96.65

respectively, due to the improvement of Acc-2 and F-score in different emotions. Thus, it can be concluded that the use of these two fusion strategies can effectively improve the accuracy of the emotion classification.

4. Conclusion

Harmful information such as extreme emotions and potential violence widely exists in multimodal data that are stored in cyberspace. Its analysis are great necessary and urgent in the field of the public information security. To this end, this study has proposed a novel TlMF method that contains both data-level and decision-level fusion. A CNN-BiLSTM neural network has been employed to produce the unimodal embeddings. Based on this, the text-audio and text-video fusion embeddings can be then generated by a tensor fusion network. Furthermore, the soft fusion method has been introduced in the decision-fusion stage of TlMF, such that the classification or prediction results can be as accurate as possible. Finally, the testing results on multimodal sentiment analysis and emotion classification tasks have confirmed that the multimodal features learned from the TlMF model can lead to state-of-the-art performance. This fully shows the effectiveness of the proposed method. Moreover, the availability of the network's components has been verified through the ablation studies.

Data Availability

The data used to support the findings of this study are found at http://immortal.multicomp.cs.cmu.edu/raw_datasets/processed data/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by Fundamental Research Funds for the Central Universities (3072020CFQ0602, 3072020CF0604, and 3072020CFP0601) and in part by 2019 Industrial Internet Innovation and Development Engineering (KY1060020002 and KY10600200008).

References

[1] L.-P. Morency, R. Mihalcea, and P. Doshi, "Towards multimodal sentiment analysis: harvesting opinions from the web," in *Proceedings of the 13th International Conference on Multimodal Interfaces*, pp. 169–176, Alicante, Spain, November 2011.

[2] M. Soleymani, D. Garcia, B. Jou, B. Schuller, S.-F. Chang, and M. Pantic, "A survey of multimodal sentiment analysis," *Image and Vision Computing*, vol. 65, pp. 3–14, 2017.

[3] W. Wang, *Machine Audition: Principles, Algorithms, and Systems*, IGI Global, Hershey, PA, USA, 2011.

[4] Y. Huang, J. Yang, P. Liao, and J. Pan, "Fusion of facial expressions and EEG for multimodal emotion recognition," *Computational Intelligence and Neuroscience*, vol. 2017, Article ID 2107451, 8 pages, 2017.

[5] D. Wu, J. Chen, W. Deng, Y. Wei, H. Luo, and Y. Wei, "The recognition of teacher behavior based on multimodal information fusion," *Mathematical Problems in Engineering*, vol. 2020, Article ID 8269683, 8 pages, 2020.

[6] S. Lee and I. Kim, "Multimodal feature learning for video captioning," *Mathematical Problems in Engineering*, vol. 2018, Article ID 3125879, 8 pages, 2018.

[7] B. Huang, F. Yang, M. Yin, X. Mo, and C. Zhong, "A review of multimodal medical image fusion techniques," *Computational and Mathematical Methods in Medicine*, vol. 2020, p. 16, 2020.

[8] S. El-Sappagh, T. Abuhmed, S. M. Riazul Islam, and K. S. Kwak, "Multimodal multitask deep learning model for Alzheimer's disease progression detection based on time series data," *Neurocomputing*, vol. 412, pp. 197–215, 2020.

[9] H. Li, A. Shrestha, H. Heidari, J. Le Kernec, and F. Fioranelli, "Bi-LSTM network for multimodal continuous human activity recognition and fall detection," *IEEE Sensors Journal*, vol. 20, no. 3, pp. 1191–1201, 2019.

[10] S.-J. Wang, H.-L. Chen, W.-J. Yan, Y.-H. Chen, and X. Fu, "Face recognition and micro-expression recognition based on discriminant tensor subspace analysis plus extreme learning machine," *Neural Processing Letters*, vol. 39, no. 1, pp. 25–43, 2014.

[11] D. Patel, X. Hong, and G. Zhao, "Selective deep features for micro-expression recognition," in *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR)*, Cancun, Mexico, December 2016.

[12] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: a survey and taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[13] S. K. D'mello and J. Kory, "A review and meta-analysis of multimodal affect detection systems," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, pp. 1–36, 2015.

[14] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, "A survey of affect recognition methods: audio, visual, and spontaneous expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 1, pp. 39–58, 2008.

[15] C. G. Snoek, M. Worring, and A. W. Smeulders, "Early versus late fusion in semantic video analysis," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, pp. 399–402, Singapore, November 2005.

[16] A. Zadeh, M. Chen, S. Poria, E. Cambria, and L.-P. Morency, "Tensor fusion network for multimodal sentiment analysis," 2017, https://arxiv.org/abs/1707.07250.

[17] Z. Liu, Y. Shen, V. B. Lakshminarasimhan, P. P. Liang, A. Zadeh, and L.-P. Morency, "Efficient low-rank multimodal fusion with modality-specific factors," in *Proceedings of the 56th Annual Meeting of the Association-For-Computational-Linguistics (ACL)*, Melbourne, Australia, July 2018.

[18] A. Zadeh, P. P. Liang, N. Mazumder, S. Poria, E. Cambria, and L.-P. Morency, "Memory fusion network for multi-view sequential learning," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New York, NY, USA, February 2018.

[19] Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal transformer for unaligned multimodal language sequences," in *Proceedings of the Conference Association for Computational Linguistics Meeting*, p. 6558, Florence, Italy, July 2019.

[20] Z. Sun, P. Sarma, W. Sethares, and Y. Liang, "Learning relationships between text, audio, and video via deep canonical correlation for multimodal language analysis," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, pp. 8992–8999, 2020.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] A. Zadeh, C. Mao, K. Shi et al., "Factorized multimodal transformer for multimodal sequential learning," 2019, https://arxiv.org/pdf/1911.09826.pdf.

[23] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks: The Official Journal of the International Neural Network Society*, vol. 18, no. 5-6, pp. 602–610, 2005.

[24] O. Melamud, J. Goldberger, and I. Dagan, "context2vec: learning generic context embedding with bidirectional LSTM," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 51–61, Berlin, Germany, August 2016.

[25] A. A. Sharfuddin, M. N. Tihami, and M. S. Islam, "A deep recurrent neural network with BiLSTM model for sentiment classification," in *Proceedings of the 2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, pp. 1–4, Sylhet, Bangladesh, September 2018.

[26] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proceedings of the 2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 273–278, Olomouc, Czech Republic, December 2013.

[27] J. A. Benediktsson and I. Kanellopoulos, "Classification of multisource and hyperspectral data based on decision fusion," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 3, pp. 1367–1377, 1999.

[28] B. Waske and J. A. Benediktsson, *Decision Fusion, Classification of Multisource Data*, Springer, Berlin, Germany, 2014.

[29] A. Mazher, P. Li, T. A. Moughal, and H. Xu, "A decision fusion method using an algorithm for fusion of correlated probabilities," *International Journal of Remote Sensing*, vol. 37, no. 1, pp. 14–25, 2016.

[30] J. Agarwal and S. S. Bedi, "Implementation of hybrid image fusion technique for feature enhancement in medical diagnosis," *Human-centric Computing and Information Sciences*, vol. 5, no. 1, pp. 1–17, 2015.

[31] G. Aiello, I. Giovino, M. Vallone, P. Catania, and A. Argento, "A decision support system based on multisensor data fusion for sustainable greenhouse management," *Journal of Cleaner Production*, vol. 172, pp. 4057–4065, 2018.

[32] A. Zadeh, R. Zellers, E. Pincus, and L.-P. Morency, "Mosi: multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos," 2016, https://arxiv.org/abs/1606.06259.

[33] A. B. Zadeh, P. P. Liang, S. Poria, E. Cambria, and L.-P. Morency, "Multimodal language analysis in the wild: cmu-mosei dataset and interpretable dynamic fusion graph," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 2236–2246, Melbourne, Australia, July 2018.

[34] C. Busso, M. Bulut, C.-C. Lee et al., "Iemocap: interactive emotional dyadic motion capture database," *Language Resources and Evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[35] H. Li, A. Shrestha, H. Heidari, J. L. Kernec, and F. Fioranelli, "A multisensory approach for remote health monitoring of older people," *IEEE Journal of Electromagnetics, RF and Microwaves in Medicine and Biology*, vol. 2, no. 2, pp. 102–108, 2018.

[36] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Jeju, South Korea, March 2014.

[37] G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer, "Covarep—a collaborative voice analysis repository for speech technologies," in *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 960–964, Florence, Italy, May 2014.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5518168, 20 pages
https://doi.org/10.1155/2021/5518168

Research Article

Anonymous Data Reporting Strategy with Dynamic Incentive Mechanism for Participatory Sensing

Yang Li[1], **Hongtao Song**[1], **Yunlong Zhao**[2], **Nianmin Yao**[3] **and Nianbin Wang**[1]

[1]*College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China*
[2]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China*
[3]*School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China*

Correspondence should be addressed to Hongtao Song; songhongtao@hrbeu.edu.cn

Received 28 February 2021; Accepted 21 May 2021; Published 1 June 2021

Academic Editor: David Megías

Participatory sensing is often used in environmental or personal data monitoring, wherein a number of participants collect data using their mobile intelligent devices for earning the incentives. However, a lot of additional information is submitted along with the data, such as the participant's location, IP and incentives. This multimodal information implicitly links to the participant's identity and exposes the participant's privacy. In order to solve the issue of these multimodal information associating with participants' identities, this paper proposes a protocol to ensure anonymous data reporting while providing a dynamic incentive mechanism simultaneously. The proposed protocol first establishes a submission schedule by anonymously selecting a slot in a vector by each member where every member and system entities are oblivious of other members' slots and then uses this schedule to submit the all members' data in an encoded vector through bulk transfer and multiplayer dining cryptographers networks (DC-nets) . Hence, the link between the data and the member's identity is broken. The incentive mechanism uses blind signature to anonymously mark the price and complete the micropayments transfer. Finally, the theoretical analysis of the protocol proves the anonymity, integrity, and efficiency of this protocol. We implemented and tested the protocol on Android phones. The experiment results show that the protocol is efficient for low latency tolerable applications, which is the cases with most participatory sensing applications, and they also show the advantage of our optimization over similar anonymous data reporting protocols.

1. Introduction

Participatory sensing is a feasible paradigm in which individuals with sensing and computing devices collectively share information to measure and map phenomena of common interest. The development profits from most mobile terminal equipment (e.g., smart phones) have multiple embedded sensors such as GPS, accelerometer, and camera. The mobile participatory sensing completes the functionality of traditional Wireless Sensor Networks (WSN) with the help of mobile phones. Moreover, participatory sensing has its advantages: more available resources and better spatiotemporal coverage from ubiquitous mobile phones, more intelligent computing from people's participation, and high scalability without specialized infrastructure. Many participatory sensing applications have been widely used in daily life and researches. These applications are mainly used in two aspects [1]: (1) people-centric monitoring applications, which include personal health monitoring applications, sport and exercises monitoring applications, social media enhancing applications, and price auditing applications and (2) environment-centric sensing applications which are interested in the phenomena around people in the community scale, for example, air quality monitoring applications, thermal and noise monitoring applications, traffic jam alerts, wireless indoor localization, and small cell network monitoring applications.

Although participatory sensing is hugely convenient for data collection through people participation, it also raises some new research challenges. The primary concern is the privacy of participants who are contributing sensing data, since collected data contains or adds some multimodal

information to implicitly connect the identities of participants which maybe expose the participant's personal or private information. When a participant submits data, such as the participant's registration information, IP or even geographic location may be associated with the participant's identity. At the same time, additional information such as incentives and reputation accompanied by data quality may also be connected to participants. For the former, it is an inevitable problem brought about by the current network structure. For the latter, with incentives as an example, unlike other traditional applications, participatory sensing is not always based on gratuitous participation. Incentives are indispensable to attract users to participate and to encourage the participants to provide sufficient sensing data and improve the quality of the sensing service. However, it is more challenging to disassociate this multimodal information from participant identities to protect participant privacy and provide incentives, especially when anonymity is provided as the protection for privacy [2]. Furthermore, anonymity allows greedy users to submit multiple data reports of the same sensing task, copy or exchange identities for more incentives, and it is difficult to detect. This not only increases the cost of collecting data but also undermines the fairness of the system, which may make participants and task requesters no longer willing to join.

Most of the researches have been proposed to address them separately, only a few researches address all of them simultaneously. In works [2–5], authors assume that the communications between users and the task requesters are anonymized (e.g., Mix networks) with the help of a trusted third party (TTP). The trusted third party is often used for providing privacy protection and incentive mechanisms and can help users hide their real identities (e.g., using pseudonyms) and transfer micropayments. However, relying on a trusted third party increases the complexity of the system and may not be able to convince participants of its credibility. For TTP-free scheme, works [6, 7] use the blind signature mechanism to sign a token and a pseudonym, and verify them by comparing for determining whether the participant is payable. However, the pseudonym and the signed token are independent. If the participants collude, they maybe exchange or duplicate others' pseudonyms for earning more payments. Our previous work [8] proposed a privacy protection scheme with incentive mechanism for mobile crowd-sensing. It uses slot reservation based on shuffle to establish a submission schedule by anonymously assigning a slot in a vector to each member in a group of participants. The shuffle leverages the participants to disrupt the order of the multiple encrypted reservation message. Data submission based on bulk transfer and multiplayer DC-nets uses members' reserved slots to transmit all thedata in an encoded vector to service provider. Finally, Blind signature ensures the incentive mechanism. The consumption of multiple encrypted reservation messages is good enough for usual MCS application, but the encryption is still a very complicated operation. The incentive mechanism ensures that participants receive payment while protecting their privacy from exposure. It will be better if it can enable each participant to get a dynamic incentive to encourage the participant to provide higher data quality.

In order to address the above issues, this paper proposes a strategy to protect participant's privacy through data reporting anonymity while providing incentives in the form of micropayments. Our contributions are as follows: (1) we present a data reporting protocol for guaranteeing anonymity based on multiplayer DC-nets and bulk transfer with an incentive strategy followed by works [9–13], and adapted to participatory sensing in Yao et al. [14] and our previous work [8]. (2) We optimize slot reservation and data submission stage for efficiency, which uses random slot selection instead of previous multiple encryption shuffles. And, the bulk transfer is simplified for this protocol maintaining the same security as our previous work. (3) And, we improve the incentive mechanism to support anonymously dynamic micropayments for different participants. It reversely uses the blind signature mechanism to carry dynamic incentives, and uses the similar mechanism as data submission to confuse the participant's identity. The theoretical analysis in this paper proves the integrity, anonymity, and efficiency of the protocol. Finally, we prove the practicality of this protocol for participatory sensing applications by implementation and performance evaluation.

The rest of this paper is organized as follows. Section 2 reviews some related work and preliminaries. Section 3 presents the system model and the adversary model. Section 3 proposes our protocol. And the protocol analysis and performance evaluation is shown in Sections 5 and 6, respectively. At last, Section 7 concludes this paper.

2. Related Works

Participatory sensing is an active research area full of open questions and challenges. The main issue is to protect the privacy of participants who contribute sensing data, because the private information of data contributors may be exposed during data submission. Some studies have been proposed to solve the privacy problem in the data publishing [15]. The first solution is to restrict the data publishing, by setting some pre-configurations to decide whether or not to publish the given data to protect the privacy of users. This requires prior interaction with the user to determine these configurations [16]. Another solution is to control the granularity of the data, that is, to reduce the accuracy of the data to a certain extent, and then sending modified data instead of accurate data to protect privacy [1, 17]. Furthermore, data perturbation [18] is also proposed to perturb individual data, while allowing the reconstruction of the original data statistics in the crowd. Although all these solutions are based on actual applications, they are a compromise between privacy protection and data quality.

Pseudonyms have been widely used to hide the true identities of participants to ensure the anonymity of data reports [19–21]. It is not enough to protect privacy by using only the pseudonyms since malicious attackers can infer the true identity of the participant from the network context and location information [22]. Due to pseudonymity limitation, k-anonymity model [23] was proposed, which aims to

reduce the data granularity until making a given report maps onto at least kk other reports. It releases data with lower precision through generalization and concealment technology, so that each record has at least the same quasi-identifier attribute value as other $k-1$ records in the data table, thereby reducing privacy leakage caused by link attacks [24, 25]. However, this scheme is usually only effective for protecting sensitive data; protecting participant identity to the level of k individuals is not the same as protecting the sensitive data, as network context and location information can still expose the user's identity.

Secure Multiparty Computation (SMC) [26] is often used to protect user privacy in distributed systems. It protects the input data privacy of mutually distrustful participants in a group without the help of the trusted third party, jointly complete certain multi-input calculations, and ensure the correctness of the output results. Existing solutions can be basically divided into two categories: oblivious transfer scheme [27] and encoding and homomorphic encryption scheme [28]. Among them, encoding and homomorphic encryption scheme is often used to solve the problem of multiparticipant collaborative transmission of data [29, 30], and there are many studies to improve the encryption or encoding [31, 32]. It can allow participants to use encrypted or encoded data to calculate, and finally obtain the calculation results, but cannot know the input data of each participant. Because using SMC cannot obtain individual data provided by participants, it is usually suitable for systems that only need to obtain statistical results. Furthermore, if malicious entities collude with stronger eavesdroppers, the privacy of participants may still be exposed.

Differential privacy protection is widely used for privacy protection. It hides the impact of a single datum by adding random disturbances to the published data, so that the calculation result does not reveal too much information of a single record in the data set [33]. At present, there are many researches focusing on the security of balancing random disturbances and privacy protection [34–36]. In order to ensure the preserving of privacy in the data collection, works [37–39] proposed several local differential privacy schemes, which allow users to perturb the data locally, and then send the results with the perturbation to the task requester. Sei and Ohsuga provide an estimation technique, which estimates the true data distribution based on the reported data on the data collector's side [40]. Ni et al. [41] propose a differentially private double auction mechanism to select participants, and the use of utility function for pricing. However, these solutions are still difficult to prevent malicious service providers from colluding with global eavesdroppers to link each datum with its contributors. Moreover, it usually disturbs all the original data excessively to ensure the anonymity of the data.

Some studies have improved the system architecture. For privacy and report integrity, Cornelius et al. [42] propose to use trusted entities to reduce the sensitivity of the data, while using a mix network to avoid network tracking. Work [43] mainly leverages an idea of blind matching of data reports and sensing tasks with the help of a third party. Although both of these solutions are good at protecting the privacy of participants, they still rely on a trusted third party and unrealistic assumptions. On the other hand, in [20, 44, 45], decentralized approaches to protect privacy are proposed. But, their technologies are still focused on specific scenarios, and it is difficult to apply to all applications.

Unlike previous works, communication anonymity techniques aim to hide the identity of the sender and receiver (in network) by removing identifying patterns from the network flow. Therefore, it preserves privacy protection during the data collection, and can submit the original individual data without sensitive information in general. Most of these techniques are based on laundering traffic through intermediates, nodes, and encryption. Dissent proposed in Corrigangibbs and Ford [10] leverage the verifiable shuffles to establish a transmission plan and then uses multiplayer DC-nets and bulk transfer for anonymous data transmission. Yao et al. [14] improve the slot reservation and data submission stage, so that participants can perform data transmission without negotiation and also preserve anonymity. The modified approach is applicable to Mobile crowdsensing (MCS), but the calculation of the slot reservation stage suffers from relatively high latency. Work [8] improved Yao's scheme, reducing the number of encryptions in the slot reservation stage while maintaining the same anonymity, but the multi-encrypted shuffle mechanisms still is a complicated operation.

In addition, most of the previous works consider privacy issues without considering incentive mechanisms. Few researches address both issues at the same time, since protecting the privacy and providing incentives simultaneously is more challenging [2], especially when anonymity is provided as the privacy-preserving mechanism. Blind signatures [46, 47] have been widely used in anonymous electronic payment systems and digital currencies. It is natural to use blind signatures to achieve privacy protection and credit. However, blind signature technology prevents the signer from linking to any specific user. Using blind signature technology alone may cause malicious users to harm other users, steal their credit lines, and spend these credit lines without being discovered. Works [3, 4] are the works that address both issues simultaneously. However, those solutions pay more attention to data content protection and anonymous incentives over communication anonymity. The authors assume that the communications between users and the SP are anonymized (e.g., Mix networks) which rely on the anonymization of the third party. Similarly, work by [48] relies on TOR anonymization network for communication anonymity which is the third party solution by just switching the problem from the task request server to the third party server. In work by [49], a multiple encryption strategy is used to keep the untraceability of pseudonymity and the network service provider is assumed to be credible and the only entity who could know the identity. Work in [43] mainly leverages an idea of blind matching of data reports and sensing tasks with the help of the third party. Gao et al. proposed an anonymous approach based on proxy ring signature to hide the identities through groups[50]. A similar scheme is used in the smart grid which has a creditable entity in the system naturally [51].

For TTP-free scheme, Li and Cao [4] present another mechanism for TTP-free scheme to use blind signature, partially blind signature, and an extended Merkle tree technique to protect user privacy and prevent abuse attacks. However, in the case of collusion between other participants and system entities, the identities of honest participants can still be exposed. Wang et al. [6] used blind signatures and differential data to hide the real identity of user and provide a reputation mechanism . The server needs to sign two sets of signatures to verify the authenticity of the information while remaining anonymous. And, Wu et al. [7] leverage blind signatures and Hash to provide a similar method. Both of them use blind signature to sign a token and a pseudonym, and verify them by comparing for determining whether the participant is payable. However, the pseudonym and the signed token are independent. If the participants collude with each other, they may exchange or duplicate others' pseudonyms for earning more payments.

3. Problem Statement

For the usual participatory sensing application, the system includes a task requester and a number of participants with their own mobile sensing device. The participants voluntarily use their devices to collect sensor data based on the task requester's request and report the data to the task requester periodically. Then, the task requester processes the reported data to extract useful statistics in varied forms according to the realistic requirements. However, the participants' sensitive information may be revealed during sensor data collection, which puts their privacy at risk. Privacy-preserving is an indispensable mechanism for the system and also obligatory to encourage volunteers. Obtaining incentives is the purpose of the participants to perform tasks, and the incentive mechanism is indispensable, which ensures the participation of users and revokes malicious participants.

The content of the reported data and the process of reporting the data to the server threaten the participants' privacy. Considering current Internet architecture, network-based traceback techniques can trace the origin of most Internet data packets [52]. Furthermore, the participants' IP addresses may expose their location, even help in inferring their real identity through analyzing of the reporting pattern. Therefore, anonymous communication is indispensable for privacy protection. This paper presents an approach to break the link between the reported data that do not contain sensitive information and the identity of the data reporter to ensure the anonymity of the communicator, so that malicious attackers cannot determine the participant submitting the data, thereby protecting privacy.

3.1. System Architecture. The goal of this paper was to hide the data reporter's identity in network and provide a way to ensure dynamic incentive mechanisms. The conception objective is to protect participants' privacy by breaking the link between data reporter identity and the reported sensing data. Most of the existing solutions could protect the privacy with incentives mechanism relying on a trusted third party. Unlike those solutions, our approach particularly does not rely on any trusted third party. In the system, participants collaborate to hide reports' sources in a decentralized model. We still envision the participatory sensing system model composed of the following entities as shown in Figure 1:

1) Task Requester (TR): TR is the participatory sensing task sponsor. It collects the reported data from a given participatory sensing application and will provide incentives for participants.
(2) Service Provider (SP): SP is the intermediary between TR and participants. It allocates the sensing task from TRs to elected participants and transfers the sensing data reports reversely.
(3) Credit Authority (CA): CA is a neutral service. It manages the real identities of other entities. Moreover, the *CA* manages also participants' and TRs' accounts, and performs credit transactions from TRs' account to the participant account who has the micropayments token.
(4) Participant: they are mobile sensing devices (e.g., smart phones) equipped with sensing, computation, and communication abilities controlled by people. The owners are volunteers who participate in the sensing tasks and report sensing data to SP before earning the incentives. We use "participant," "user," or "group member" interchangeably in the remainder of this paper.

3.2. Adversary Model. For security and privacy, this protocol requires that each group of participants include at least two honest participants to make it impossible to distinguish between two data reporters, and these two participants could not collude with others or global eavesdroppers. Besides, all entities could be honest-but-curious in this system, in which each entity may try to learn about other participants passively, but faithfully follows the protocol. In other words, all entities follow protocol specifications but try to get some private information about other participants. This protocol allows global and passive eavesdroppers to exist, who can monitor all the communications in the network. This consideration is realistic considering network providers and government agencies. Furthermore, system entities can collude with each other and even with powerful eavesdroppers trying to get the link between participants to reported data.

For obtaining incentives, participants can try to earn more rewards than that of the sensing report's worth. For example, the participant uses data duplication attack, or uses different identities to report the same sensing data content, even report faked data content. We also assume that participants may try to steal others' certificates, use an invalid certificate, or use a valid certificate to get rewards repeatedly.

3.3. System Assumptions. In this system, all entities have their own private/public key pair, including each participant P_i having (sk_i, pk_i), TR (sk_{TR}, pk_{TR}), SP (sk_{SP}, pk_{SP}), and CA

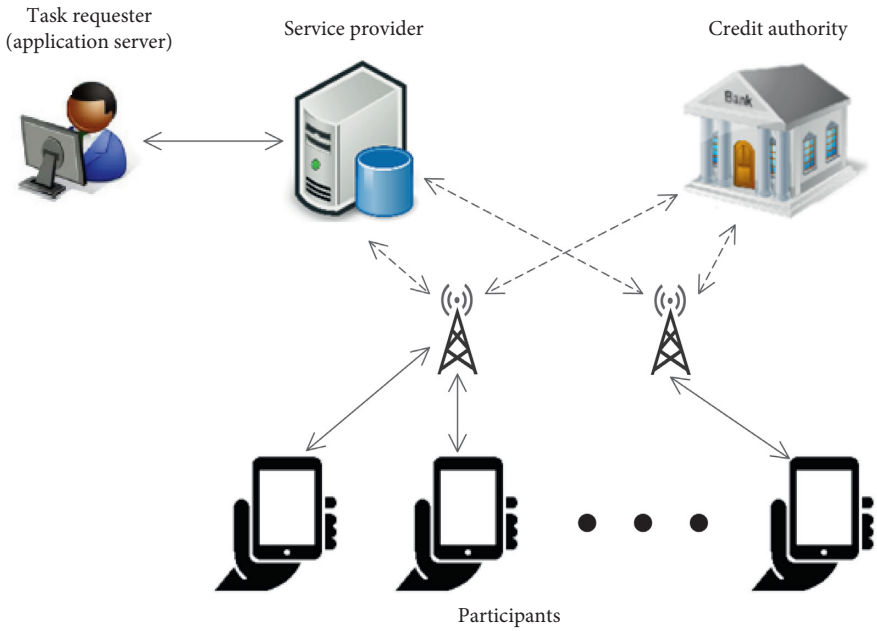


Figure 1: The system model with different entities.

(sk_{CA}, pk_{CA}). And the CA has another private/public key pair (sk^R_{CA}, pk^R_{CA}) for encrypting the tag of incentives. Each entity can generate the key pair itself or apply for a key pair from a certification authority. The members in the same group can communicate with each other in a peer to peer mode, and all communications between different entities are encrypted. Finally, our discussion will be in one group causing the functions of every group to be in the same manner.

4. Protocol Design

The goal of communication anonymity techniques is to confuse the identities and data through multiple participants submitting data together. Most of them use the idea of Chaum's Mix network and DC-nets. Mix networks are mainly designed to work in a centralized fashion with the help of fixed servers; DC-net hides the real data covered by negotiated masks. Nevertheless, this idea has been extended in bulk transfer in Dissent [10], which makes it feasible in a distributed way. The anonymous data reporting protocol in this paper (called OADR scheme for short in the remainder of the article) also leverages the DC-net and bulk transfer followed by works [8–14]. In this paper, we firstly modify bulk transfer protocol. Then, all the processes of OADR scheme shows in Section Protocol Description. The related notations used in the remainder of the article are summarized in Table 1.

4.1. Modified Bulk Transfer. The bulk transfer protocol aims to submit a random arrangement of N participants' data in a vector of N slots. We assume that participants $P_1, P_2, \ldots, P_N$ initially hold an amount of messages $\text{msg}_1, \text{msg}_2, \ldots, \text{msg}_N$ willing to report. Each participant P_i knows its own slot position in the vector but does not know the other participants' slots. The lengths of the whole vector and each slot are published. Each pair of participants (P_i and P_j) shares a common secret seed S_{ij}, and note that $S_{ij} = S_{ji}$.

First of all, P_i generates a bitstream C_i^{msg} with the length of the whole vector (let beL_v). C_i^{msg} could be split into N slots which are filled in P_i's slot with msg_i, and the bits in the other slot are all zero (the placeholder #), i.e.,

$$C_i^{\text{msg}} = \overbrace{00\cdots0}^{L_b} \,|\text{msg}_i|\underbrace{00\cdots0}_{L_v}, \tag{1}$$

where | denotes concatenation and L_b is the demanded bits of the other slots ahead of P_i's. Then, P_i continues generating $N-1$ pseudorandom bitstreams C_{ij}, all with the length L_v and based on the seed shared with other participants as

$$C_{ij} = \text{PRF}\{L_v, S_{ij}\}, \quad j \neq i, \tag{2}$$

, where the j is in [1, N] corresponding to all participants beside itself. The PRF is the signification bits of length L_v generated by a pseudorandom function taking S_{ij} as seed.

After that, P_i generates vector C_i using C_i^{msg} XOR all C_{ij}, i.e.,

$$C_i = C_i^{msg} \oplus C_{i1} \oplus C_{i2} \oplus \cdots \oplus C_{i(i-1)} \oplus C_{i(i+1)} \oplus \cdots \oplus C_{iN}. \tag{3}$$

The bitstreamsC_i for $i \in [1, N]$ are sent to the service provider, where all of them are XORed.

$$C = C_1 \oplus C_2 \oplus \cdots \oplus C_N = \text{msg}_{\pi N(1)}|\text{msg}_{\pi N(2)}|\cdots|\text{msg}_{\pi N(N)}. \tag{4}$$

Finally, the service provider obtains a concatenation of all data in the vector C (see Figure 2), where $\text{msg}_{\pi N(i)}$ is the random permutation in $\text{msg}_1, \text{msg}_2, \ldots, \text{msg}_N$. The service provider has been sent all the messages from participants. The

Table 1: Summary of notations.

Notation	Description
P_i	The i^{th} participant in order (member, user)
S_{ij}	The secret seed between the i^{th} participant and the j^{th} participant
SRM_i	The slot reservation message for the i^{th} participant
$\vert$	Concatenation operator
#	The placeholder
PN_i	Random secret pseudonym of the i^{th} participant
$\pi_{i-1}(j)$	The j^{th} element in order after the random permutation of $i-1$ elements
n_{R_j}	The nonce for the j^{th} slot, earlier agreed on its creation rule
$\mathrm{PRF}\{L, x\}$	Pseudorandom generation function based on seed x, the length of the generated pseudorandom is L

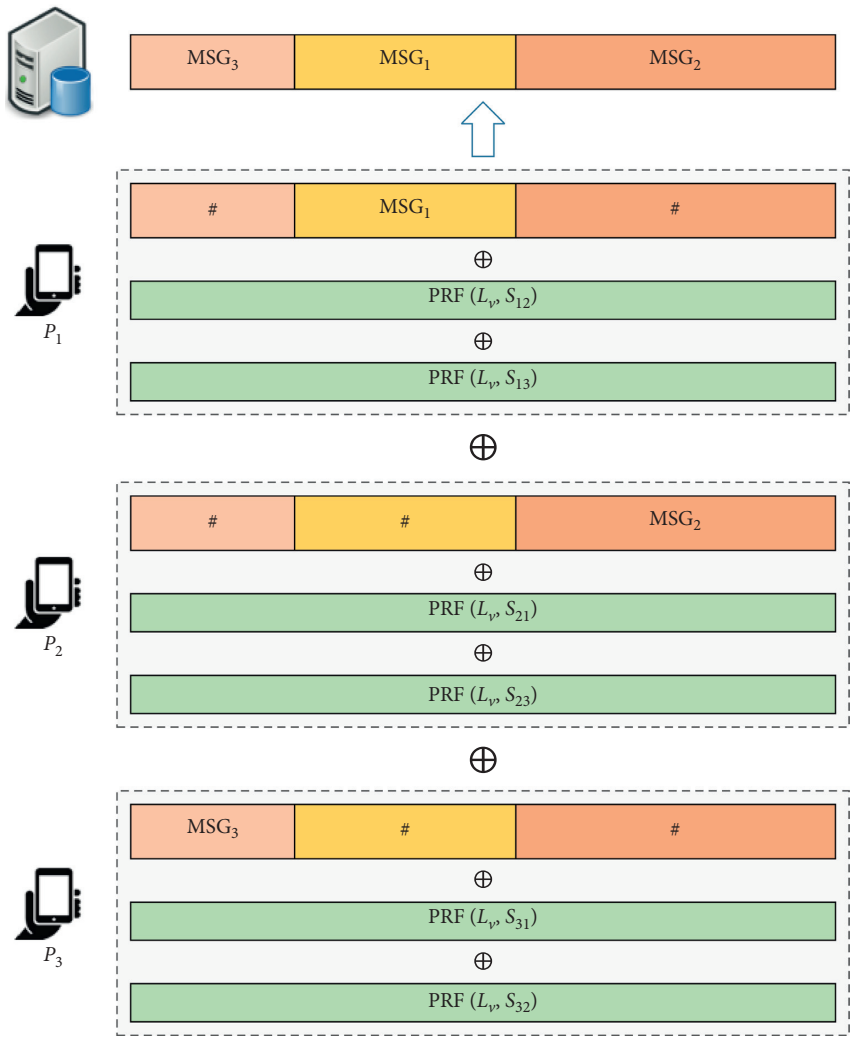


Figure 2: Illustration of modified bulk transfer for group of 3 participants using permutation [1–3].

link between the message and its own is only known by the owner if the order of the permutation is shuffled. Comparing with the original bulk transfer protocol, this reduces the time of pseudorandom generating maintaining the same anonymity.

4.2. Protocol Description. This protocol is mainly based on work by Yao et al. [14] and our previous work [8] (called Yao's scheme and ADR scheme, respectively); the differences among Yao's ADR and OADR scheme are that: in Yao's scheme, each participant receives and sends N slot reservation messages with N times encryption and decryption. In the ADR scheme, each participant gets only its predecessors' slot reservation messages and encrypts its own slot reservation messages with successors' public keys, and decrypts once for each participant during the whole slot reservation stage. For OADR scheme, it uses slots random selection in bulk transfer instead of the encrypted

pseudonyms permutation to reserve the slots, thereby reducing the latency of the slot reservation stage, and for the data submission, the process is simplified due to the modification of bulk transfer. Furthermore, the incentive mechanism supports dynamic rewards and enhances anonymity based on blind signature scheme and bulk transfer. The OADR scheme is formalized into five main sequential stages as ADR scheme including: setup stage, slot reservation stage, token requesting stage, data submission stage, and token deposit stage. Except for the setup stage, all the stages leverage the modified bulk transfer mentioned (see the Section "modified bulk transfer") to submit data.

4.2.1. Setup Stage. For the new participants, they need to register to the CA and the SP using their real identities. Considering the sensing data reports may involve participants' location or the task may request data from certain locations, and it is best to select participants in the same group in the same geographic area. There are many researches to prevent the spatial information fromthreatening user privacy. For example, work by [53] shows a way to divide the space into subregions, and the user's location is hidden in l possible sub-regions. Without loss of generality, the N group members are denoted as $P_1, P_2, \ldots, P_N$.

At last, the group members exchange their public keys pk_i, such as using a digital certificate. Using the public key, each participant P_i shares a secret seed and a random function with every other participant P_j for generating the secret S_{ij} between P_i and P_j, which makes S_{ij} the same as S_{ji}, which is only known between them and could generate pseudorandoms for multiple rounds. Furthermore, the format and length of the slot reservation message, and the number of redundancy slots used in the slot reservation stage should be agreed upon by all the participants and SP.

4.2.2. Slot Reservation Stage. The goal of this protocol is to submit the reported data from a whole group of members together, in which the permutation of the different participants' data is disordered to break the link between the reporter and corresponding data. In other words, the participant P_i just knows its own position of the data but the others' position is ambiguous to P_i. In the slot reservation stage, the position of each participant will be determined. Unlike Yao's scheme and ADR scheme using random permutation of encrypted pseudonym to determine the submission order for ensuring anonymity, the slot reservation stage in OADR is optimized which leverages the idea of DC-net and bulk transfer for selecting the slot in redundancy slots. The whole process of the slot reservation stage for a group of three members is illustrated in Figure 3.

At the beginning of this stage, each participant P_i needs to generate a λ-bits fresh random secret pseudonym denoted as PN_i. Obviously, the pseudonym is only known by the participant itself. As the pseudonym is randomly generated, the collisions among all group members are possible. The collisions' probability reduces with the growing of λ and the decreasing of the number of participants in one group. Nevertheless, OADR no longer needs to care about whether the SRM itself will conflict, it only needs to focus on whether the slot conflicts.

Once the pseudonym generation was completed, each participant P_i prepares a slot reservation message (SRM). In addition, the format and length of the slot reservation messages should be decided early by all participants and SP in order to produce the unique format and length SRM. Otherwise, the vector of SRM will not be able to be recognized or identified. The basic SRM is shown in the following equation:

$$\mathrm{SRM}_i = \langle \mathrm{PN}_i | L_i \rangle, \tag{5}$$

where L_i is the length of data with the length of an encrypted token used later for incentives (blinded token) which P_i will report to the task requester (TR) anonymously. Obviously, the data length (L_i) should be the length of the encrypted data with $\mathrm{pk}_{\mathrm{TR}}$ if the data content need to be kept confidential from SP.

The next step is that each participant P_i transfers its SRM_i to SP by bulk transfer. Each participant P_i uses each secret seed S_{ij} to generate $N-1$ pseudorandom bit stream vectors C_{ij} of length $M \cdot L_{\mathrm{SRM}}$, as shown in the following equation:

$$C_{ij} = \mathrm{PRF}\left\{M \cdot L_{\mathrm{SRM}}, S_{ij}\right\}, \quad i \neq j, \tag{6}$$

where PRF represents a pseudorandom vector of length $M \cdot L_{\mathrm{SRM}}$ generated by a pseudorandom function with S_{ij} as a secret seed. L_{SRM} is the length of each SRM. M is the total number of slots, which is greater than the total number of participants N. Note that M should be negotiated and decided by all participants and SP when deciding the format and length of SRM. Therefore, the vector can be divided into M slots of equal length, enough to accommodate all participants' slot reservation information, and some unused slots will remain. Then, each participant P_i selects a random slot (let the position be pos_i) to hold its SRM and generatesC^{poi_i}, as shown in equation (7):

$$C^{\mathrm{pos}_i} = \#_1 | \#_2 | \cdots | \#_{pos_i - 1} | \mathrm{SRM}_i | \#_{\mathrm{pos}_i + 1} | \#_M, \tag{7}$$

where $\#_1$ represents the placeholder of the i^{th} slot (that is, all zeros), $i \in [1, N]$ and $i \neq \mathrm{pos}_i$. Finally, P_i XOR all C_{ij} and C^{poi_i} to complete the generation of vector C_i:

$$C_i = C^{\mathrm{pos}_i} \oplus C_{i1} \oplus C_{i2} \oplus \cdots \oplus C_{i(i-1)} \oplus C_{i(i+1)} \oplus \cdots \oplus C_{iN}. \tag{8}$$

After that, each participant P_i sends C_i to the service provider (SP) by using bulk transfer protocol. Finally, the SP

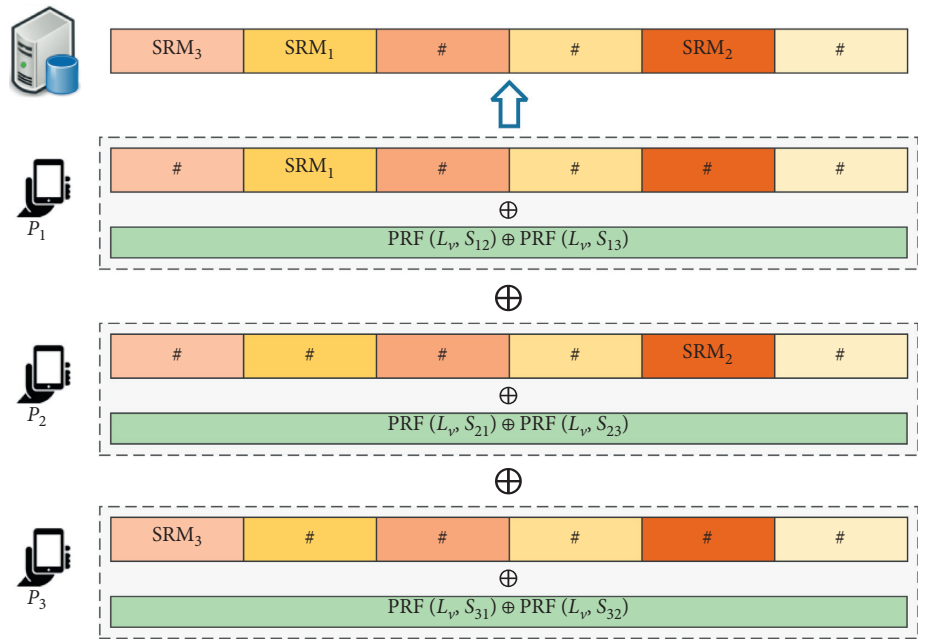


Figure 3: Illustration of a slot reservation by six slots for a group of three participants.

obtains the slot reservation vector C by performing XOR operation on all received C_i, as equation (9) shows:

$$C = C_1 \oplus C_2 \oplus \cdots \oplus C_N = \underbrace{\#|\cdots|\mathrm{SRM}_{\pi N(1)}|\cdots|\#|\cdots|\mathrm{SRM}_{\pi N(N)}|\cdots|\#}_{M}, \tag{9}$$

where $\mathrm{SRM}_{\pi N(i)}$ is the i^{th} element in the random permutation of $\{\mathrm{SRM}_1, \mathrm{SRM}_2, \ldots, \mathrm{SRM}_N\}$. SP gets the vector C and checks the number of SRMs in the vector to confirm whether there is a slot collision (i.e., whether multiple participants have selected the same slot), and then publishes C. Each participant checks whether the SRM of its selected slot in the published vector C is correct. When a time slot collision is found, the slot reservation stage needs to rerun again from the step of pseudorandom generation (starting from equation (6)). Each participant generates a new round of pseudorandom and finally builds a new vector C. For the slot of the participant P_j that does not collide (i.e., P_j checks whether its SRM_j in the vector C is correct), P_j places its SRM_j in the same time slot as before; on the contrary, the slot that has collided participant P_k needs to select another unused slot to carry its SRM_k. If all participants do not find their own SRM errors but the total number of SRMs is still insufficient, all participants must reselect their own slots. Then, all participants sends the newly generated vector C_i to SP to check again. The process of SP checking time slot collision will repeat until all participants confirm that their SRMs are correct, and the stage is successfully completed. After that, each participant P_i determines its own reserved slot, which is the position of its SRM_i, without a placeholder. P_i calculates the data length declared by all SRMs to confirm the starting position of its reserved slot in the data submission stage and the total length of the data vector. Note that when the participant P_i calculates its time slot, it needs to ignore the placeholder and only calculate the length declared in all SRMs.

In addition, the slot collision is incidental when the redundancy slots are not many. The probability of collision is shown in the following equation:

$$P_{M(N)} = 1 - \frac{M}{M} \cdot \frac{M-1}{M} \cdot \cdots \cdot \frac{M-(N-1)}{M} = 1 - \prod_{k=1}^{N-1}\left(1 - \frac{k}{M}\right). \tag{10}$$

Since each redundancy slot brings extra XOR operations, the increase of redundancy slots will reduce the probability of collision as well as increase the operation time dramatically . In the case of 800 slots ($M = 800$) for 40 members ($N = 40$), the probability of collision is 0.63. But with 100 slots for 5 members, the probability of collision is less than 0.1. Therefore, it is suggested to divide participants to multiple groups instead of one group for many participants in order to keep a low probability of collision for fewer slots. On the other hand, the collision is inevitable for a large number of participants. But, fortunately, the probability of collision is usually low in the second round of slot

reservation. According to the experiments, the average probability of collision for the second round is less than 0.14 even for $M = 200$ and $N = 40$.

4.2.3. Token Requesting Stage. Once the slot reservation stage is completed, participants request micropayment tokens from the Credit Authority (CA). Firstly, participants generate N secret seeds (denoted as PI_i) and encrypt the pseudonyms (let be $\mathrm{PI}_i^{\mathrm{pk_{CA}}}$). In this stage, participants' identities and tokens need to be unlinked; thus, all participants send $\mathrm{PI}_i^{\mathrm{pk_{CA}}}$ to CA together using the similar approach as the slot reservation stage to transfer the tokens. At the same time, one of the participants P_k sends the vector with the task id, session number, and round number to CA. CA confirms the session number and the round number with the SP. This slot reservation for tokens request still needs to protect from collision. Therefore, CA publishes the received vector containing encrypted pseudonyms for confirmation by participants as the slot reservation stage does, until all participants confirm the submissions. At last, CA removes all the placeholders and publishes the vector for determining the order token transfer.

After each participant ensures its own slot, CA generates N fresh random integers of δ-bits called incentive tokens, denoted as τ. The token is generated based on the hash function by the seed including random secrets, task identity, session number, round number, participant identities, and the current time, in order to make sure that the token is unique and fresh for each request. Next, CA decrypts all secret seeds $\mathrm{PI}_i^{\mathrm{pk_{CA}}}$ and uses them to generate N pseudorandoms $\mathrm{PRF}\{L_\tau, \mathrm{PI}_i\}$. After that, CA xor τ and $\mathrm{PRF}\{L_\tau, \mathrm{PI}_i\}$ are put it into the corresponding slot to build the final token vector V_T, as shown in equation (11).

$$V_T = \langle \mathrm{PRF}\{L_\tau, PI_{\pi N(1)}\} \oplus \tau_{\pi N(1)} \rangle | \langle \mathrm{PRF}\{L_\tau, PI_{\pi N(2)}\} \oplus \tau_{\pi N(2)} \rangle | \cdots | \langle \mathrm{PRF}\{L_\tau, PI_{\pi N(N)}\} \oplus \tau_{\pi N(N)} \rangle. \tag{11}$$

Finally, CA publishes the vector V_T, and the tokens are fetched back by each participant from its slot. The participant P_i uses PI_i to generate the pseudorandom and then XOR the pseudorandom with the fetched tokens $\mathrm{PRF}\{L_\tau, \mathrm{PI}_i\} \oplus \tau_i$ to obtain real token τ_i.

Next, the participant P_i collects a random secret integer r_i which needs to satisfy equation (12):

$$\begin{cases} 1 < r_i \le 2^\delta - 1, \\ \gcd\left(2^\delta, r_i\right) = 1. \end{cases} \tag{12}$$

After that, P_i generates a series of blind tokens using the secret factor r_i and different TR's incentive public keys based on the RSA blind signature scheme without loss of generality:

$$\tau_i^* = \tau_i \cdot (r_i)^{\mathrm{pk_{TR}}} \mathrm{mod} n_{\mathrm{TR}}. \tag{13}$$

For simplicity, τ_i^* is called the blind incentive token of the participant for the rest paper.

4.2.4. Data Submission Stage. After the slot reservation stage, each participant P_i has a slot position, $\mathrm{pos}_i \in [1, N]$. And P_i holds the sensing data and blind token $m_i | \tau_i^*$ (mt_i for short) of length L_i willing to report. Due to the order of the slots and the lengths of the data being determined, all participants could use bulk transfer protocol to report the data to SP directly. Finally, SP obtains the vector C:

$$C = mt_{\pi N(1)} | mt_{\pi N(2)} | \cdots | mt_{\pi N(N)}. \tag{14}$$

SP publishes the received data C for each participant and checks its own data including sensing data and the blinded token. Next, each participant reports a flag bit to the SP to confirm the data using the same data submission protocol. The flag could be a decided bit like the flag is set to 1 when data content is correct; otherwise, the flag bit is zero. The SP receives the confirmed report vector and checks whether there is any error. If all the data are correct, SP sends the data vector C to the task requester (TR). On the contrary, if there is one or more zero in the flags vector, the SP needs to execute the error recovering step for repairing the data.

The error recovering step requires all participants to repeatedly submit the reported data, and the pseudorandoms used in the process need to be regenerated. Considering about malicious members, some blame mechanisms should be run if the error is detected multiple times, like work by [10]. In addition, selective ordering can be used to detect malicious members.

4.2.5. Token Deposit Stage. After the data submission stage, the TR receives the final message vector C from the SP. For the data mt_i in each slot, TR decrypts the sensing data $m_{\pi N(1)}$, then checks the freshness and content of the data with the task agreed terms. According to the quality of the data, TR decides the amount of the incentive for the sensing data m_i (run some data evaluation mechanisms to estimate the data). Then, it collects a random secret integer for representing the amount where the process is similar to r_i collection, denoted as R_i. *TR* signs the incentive token (τ_i^*) with the private key of the corresponding level $\mathrm{sk_{TR}}$ as shown in the following equation:

$$\begin{aligned} \mathrm{sign}\left(\tau_i^*\right) &= \left(R_i^{\mathrm{pk_{CA}}} \cdot \tau_i^*\right)^{\mathrm{sk_{TR}}} \mathrm{mod}\, n_{\mathrm{TR}} \\ &= \left[R_i^{\mathrm{pk_{CA}}} \cdot \tau_i \cdot (r_i)^{\mathrm{pk_{DC}}}\right]^{\mathrm{sk_{TR}}} \mathrm{mod}\, n_{\mathrm{TR}} \\ \mathrm{sign}\left(\tau_i^*\right) &= \left(R_i^{\mathrm{pk_{CA}}} \cdot \tau_i\right)^{\mathrm{sk_{TR}}} \cdot r_i \mathrm{mod}\, n_{\mathrm{TR}}. \end{aligned} \tag{15}$$

Meanwhile, TR encrypts R_i by CA's incentive publish key (denoted as $R_i^{\mathrm{pk_{CA}}}$) and concentrates it with $\mathrm{sign}(\tau_i^*)$. Next, it

concentrates all $\text{sign}(\tau_i^*)|R_i^{\text{pk}_{\text{CA}}^R}$ to be a vector according to the order of slot reservation. The vector is sent back to SP for publishing. If the vector needs to remain confidential from other entities, TR could use a part of the data to mask each slot of the feedback vector as CA sends tokens to participants in token request stage.

Each participant P_i fetches its own signature tokens from the published vector. P_i will unblind signature incentive token as follows:

$$\begin{aligned}\text{sign}(\tau_i) &= \text{sign}(\tau_i^*) \cdot r_i^{-1} \bmod n_{\text{TR}} \\ &= \left(R_i^{\text{pk}_{\text{CA}}} \cdot \tau_i\right)^{\text{sk}_{\text{TR}}} \cdot r_i \cdot r_i^{-1} \bmod n_{\text{TR}} \\ \text{sign}(\tau_i) &= \left(R_i^{\text{pk}_{\text{CA}}} \cdot \tau_i\right)^{\text{sk}_{\text{TR}}} \bmod n_{\text{TR}}.\end{aligned} \tag{16}$$

Finally, all the participants ally to request micropayment from CA by the incentive deposit message, which uses the same order in token requesting stage to generate the vector that each slot contains encrypted message $\tau_i|\text{sign}(\tau_i)|R_i^{\text{pk}_{\text{CA}}^R}|\text{account}_i$ (one of participants sends the vector to CA with task id, session number, and round number). The CA decrypts and removes R_i from $\text{sign}(\tau_i)$ as shown in equation (16), checks again the token freshness with round and session number, signature validity, and whether the token τ_i belongs to the corresponding slot. Then, CA performs a micropayment transaction from TR's account to the corresponding member account after confirming the specific incentive represented by different R_i. The participant's account (account_i) should not be associated with the participant's real identity. Furthermore, it is suggested that the participant changes the account when the members change, or the CA uses an untraceable digital currency payment (such as "Monero") to preventthe participant from being tracked when using the incentive. But this is not the focus of this paper.

In addition, the protocol execution may include multiple sessions where each session may include multiple rounds (Figure 4). A round represents the data submission stage using the same transmission plan decided in the previous slot reservation. In contrast, a new session requires execution of the slot reservation stage. The token requesting and deposit stage is independent of the data submission stage; the order of token transmission can also be performed in multiple rounds and sessions using the same principle. Furthermore, the setup stage only needs to run once for a group if the members do not change.

5. Protocol Analysis

This protocol is used to ensure the anonymity of data reporting with incentive and reputation mechanism. In this section, the analyses of the protocol will be discussed, including anonymity, integrity, and efficiency.

5.1. Anonymity. The protocol preserves anonymity in a group containing at least two honest participants, where other malicious members colluding with system entities and global eavesdroppers cannot determine which honest members submitted which sensing data with nonnegligible probability.

Firstly, we need to analyze the unobservability of OADR's bulk transfer. In the bulk transfer of OADR, the real data are hidden under the pseudorandom number generated by S_{ij} as the secret seed. If a malicious attacker wants to get the real data, it must know the pseudorandom number. Part of the pseudorandom number is mixed with the real data (by XOR operation), and only P_i and P_j know S_{ij} and the corresponding pseudorandom generation function. When the pseudorandom number generation function and the secret seed are uncertain, it is very difficult to obtain a complete pseudorandom, which is guaranteed by the security of the pseudorandom generation function. Therefore, the bulk transfer protocol of OADR is unobservable.

Next, we analyze the anonymity in each stage. In the setup stage, the participant's real identity is used to prevent the same user from faking to be multiple participants for earning the incentive. But the real identity will not appear with the data submission, so it does not expose the sensing data anonymity. On the other hand, the key pair of each participant is binding to its real identity, which the certification authority can discover. However, the key pair is only used for exchanging secrets. The anonymity in data submission does not rely on public keys but relies on the bulk transfer. Even some entities can discover the identity of participants, but still cannot link their submitted data. In addition, the certification authority could verify the identity to reduce the risks of impersonated multiple members. In slot reservation and data submission stages, the modified bulk protocol still ensures unobservability, which is stronger than unlinkability and anonymity. Before proofs, we repeat the definition of semantically secure cryptosystem, which has been stated in the previous work.

5.1.1. Definition. A cryptosystem is semantically secure if any probabilistic, polynomial-time algorithm that is given the ciphertext of a certain message m (taken from any distribution of messages), and the message's length, cannot determine any partial information on the message with probability nonnegligibly higher than all other probabilistic, polynomial-time algorithms that only have access to the message length (and not the ciphertext) [54].

In other words, for two encrypted equal-length messages m_0, m_1, an adversary knowing m_0, m_1 cannot determine which is the cipher corresponding to m_0 and which one corresponds to m_1 with a probability significantly greater than 0.5.

The random slot reservation selected by the participant ensures that the position is only known to the participant. Because it is selected at random, the probability of the participant selecting any slot is $1/M$ (M is the number of slots). Therefore, if a malicious adversary wants to know the position chosen by the participant, it needs to obtain the position of the real data sent by it. Suppose an attacker controls all members except the two members whose submitted data are to be anonymized, at least one seed between the two honest members should be unknown. The SRM of an honest member must be encrypted with a one-time

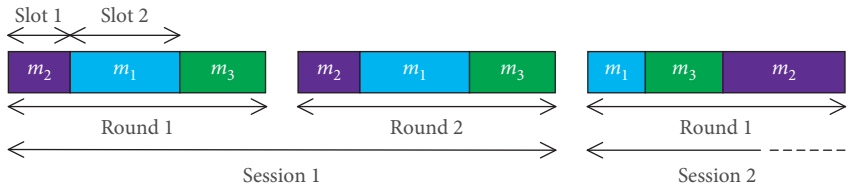


Figure 4: Example of transmission schedule for three anonymous group members.

pseudorandom generated by a seed only known to the honest member. In this case, the unobservability of the participant's SRM is guaranteed through XOR of the pseudorandom. If the malicious attacker wants to know the slot selected by the participant, it must know the pseudorandom generated by the honest participants. As analyzed above, the attacker cannot obtain a complete pseudorandom, when the pseudorandom number generation function and the secret seed are unknown. Similarly, the secret seed cannot be calculated. The attacker cannot intercept messages sent by honest participants, and tries to remove the pseudorandom to determine which slot the participant puts its SRM into. Even if the attacker compares the bit streams of two honest participants, it can only find that the two slots are different, but still cannot determine which participant selected which slot. If it is a slot collision that causes the re-run time slot reservation, the pseudorandom will be regenerated, and the attacker still cannot cancel the pseudorandom bit by comparing the pseudorandom in the two slot reservation stages. The only way to get the message content is to get all the ciphertexts generated by all members. Therefore, the attacker only knows which slots these honest participants are using but cannot determine which slot corresponds to which honest participant and the probability is greater than $1/K$ (K is the number of honest participants). In the end, the attacker cannot learn anything about the arrangement from the message. Since slot reservation ensures the anonymity of the arrangement of honest participants, an attacker cannot link participants and their reported data during the data submission stage.

In the token request stage, the anonymity of the participant is not exposed due to unknown relations between the token and the participant going by bulk transfer protocol. Furthermore, work in Lucre [55] for anonymous tokens can be adapted to overcome some attacks, such as private key extraction attacks. When the participant retrieves the token, the unobservability of the token is guaranteed by XOR of the token with a pseudorandom. The secret seed and pseudorandom generation function are only known to the participants in the corresponding slot and CA. Therefore, a malicious attacker cannot learn the content of the token through observation, and thus cannot pretend to be the participant by the token. The participants themselves can easily calculate the corresponding pseudorandom and obtain real tokens. Furthermore, the methods of keeping the confidentiality of the token are various, pseudorandom generation and XOR operation are more efficient.

In the last stage, the anonymity is mostly derived from the blind signature scheme. Here, the blind signature mechanism is used twice, once to verify the legitimacy of the token and then to carry specific incentives. Blind signature gives a signer the ability to sign a document without knowing the contents. Moreover, the signer cannot determine when or for whom he signed a given pair of signatures, even though the signer can verify whether the signature is valid. Blind signature is mostly used to preserve the anonymity of the signature requester. On the other hand, TR and CA possibly collude with each other. A possible attack is to sign different tokens or use dynamic incentives for the purpose of de-anonymizing members' contributions. But, the final beneficiary account or digital cash is irrelative to the user's identity, which is derived from bulk transfer and one-time account. Therefore, TR and CA with other malicious members just link the data to the account and not the user's identity. Even SP is malicious to select a certain honest member to participate in multiple sensing tasks with different other members; however, no entity can obtain the relation between the member and the data if the member changes its account.

This protocol preserves the anonymity of K honest participants in each group. It establishes an out-of-order submission schedule by using random slot selection; then, it submits the out-of-order encoded data by all the participants in a group, so that the relationship between the data and the identity of the data contributor is cut off, and the identity of each participant is hidden among the K honest participant. Therefore, any malicious entity cannot distinguish any two honest participants with a nonnegligibly high probability, i.e., it satisfies the semantic security of the cryptosystem. This anonymity is irrelevant to the number of honest participants. Even if there are only two honest participants, a malicious entity still cannot determine who sent which data. However, more honest participants can significantly reduce the probability of a malicious entity identifying the participant.

In addition, if all honest participants have submitted the same data, the SP or TR can determine the content of the data and the identities of all honest participants. Therefore, this protocol is not suitable for applications where the diversity of the reporting data is too low. However, this situation is not common for participatory sensing applications, especially when data need to be collected continuously. Furthermore, if the diversity of the data is low, a data diversity check should be conducted before submitting the data. First send a message digest of the data that will be submitted; the SP will publish all the message digests and all participants will check them, and then consider whether to submit the data. However, adding the diversity check will lead to an extra latency of sending data each time. Or, add privacy protection for data publishing, such as local

differential privacy protection. In this case, this protocol may not be able to submit the original individual data.

5.2. Integrity. The protocol is based on an honest but curious adversary model. Therefore, all system entities and participants must follow the protocol scheme but can passively try to destroy the anonymity of participants. If all members run the protocol scheme exactly, in the first stage, each participant has a secret seed with and the public key of each other participant, and also public keys of TR, SP, and CA.

In the slot reservation stage, each participant selects a slot randomly to reserve the slot for data submission. They update the slot reservation message encrypted with a series of one-time pseudorandoms generated from the seed sharing with other members, and then, the XOR operation will cancel all pseudorandom bits and the slot reservation message is anonymously recoverable. The process is the same as performing in the data submission stage. In the token request stage, the CA generates random nonspecial tokens and keeps records in a table associated with the round number and the session number. Furthermore, CA links the slot to the token but cannot know who obtains which token. In the last stage, the TR blindly signs members' tokens; the unblind signature can be easily calculated by the token owner but is difficult for other members. We reversely use the blind signature mechanism to carry a specific amount of incentives, which makes it difficult to forge. Because it cannot be observed, even if participants collude, it is difficult to disrupt the operation of the system by exchanging encrypted incentive tags. Finally, each member gets its incentive just by checking the availability of the token and the correctness of the slot. In addition, the token is kept secret and since it is randomly generated, it is hard to guess [40]. On the other hand, the slot reservation stage and the data submission stage use similar strategies. The former is to establish a verifiable transmission plan to facilitate subsequent anonymous data transmission. If all participants submit the data of the same length only once, it is possible to use the slot reservation stage strategy for data transmission. For larger data or data that are submitted multiple rounds, determining the transmission plan can effectively reduce the extra overhead caused by the submission of redundant data and the collision of slots.

If the system entity or participant is dishonest, that is, it can perform without following the requirements of the protocol, the system is vulnerable to attack or even unable to operate, but it is easy to detect many malicious behaviours, e.g., the protocol can detect double use of any token, even the if the participant uses tokens of others or puts its token to another slot, because these operations lead the token verification to fail by the principal of blind signature mechanism. On the other hand, the bitstream in bulk transfer is possible to be forged or members can put a malicious message in an inadequate slot. These malicious behaviours are still easily detectable. Selective order could remove a part of members including the malicious ones. Similarly, misbehaviour of TR or SP can be lead to participants' leaving. Furthermore, participants may accidentally submit wrong data, causing TR or SP to be unable to obtain the information submitted by all participants. This situation is the same as members put a malicious message in an inadequate slot. These errors will be detected during the error recovery stage after all data have been submitted. This type of problem can usually be solved by resubmitting the data. If there is indeed a malicious member, the blame mechanisms in work [10] or selective order can be used to exclude the malicious participant.

In addition, members leaving is a serious problem in all distributed anonymity preserving schemes, whether it is a participant who accidentally exits the system or goes offline halfway. Generally, a member who leaves for a long time requires the restart of the protocol; the maximum leave time must be estimated early and set as a parameter depending on the application (participatory sensing application). The whole stage will be halted if a member leaves before completing its job; depending on the participatory sensing application, the group decides whether to wait for the member or to restart the protocol without considering it. All the stages use the bulk transfer except the setup stage; a member leaving means that the whole group submitted data is incomplete. In that case, the SP depending on participatory sensing application decides to wait for that user data or to restart the protocol.

5.3. Efficiency. In the setup stage, each participant needs to generate N–1 secret seeds to share with each other member, which is the same as work [8].

In the slot reservation stage, each participant needs to generate N–1 pseudorandom bits streams. And then, this protocol performs N^2 XOR operations instead of previous $(N \cdot (N-1))/2$encryptions and $(N \cdot (N-2))/N$ decryptions to permute the slot randomly. However, the traffic overhead is also increasing $(M-1) \cdot L \cdot N$ from $N^2 \cdot L$, where M is the number of redundancy slots and the L is the SRM predefined length (all SRMs must have the same length). Even if the traffic overhead increases, the efficiency improvement brought about by replacing encryption with fewer XOR operations is huge. And, in reality, each participant could transfer the message at the same time, leading to lower transfer latency unlike when each participant needs to wait for the previous participant's message to decrypt and permute in previous work. It is similar to the data submission stage; the number of pseudorandom bits stream generations is reduced from $N \cdot (N-1)$ to $N-1$ for one participant, but the number of bits XOR is the same even if the times of XOR operations reduce to N. Once received by SP, it needs to compute N times the XOR operations for $\sum_{i=1}^{N} L_i$ bits.

In the token request stage, the CA needs to execute a random token generation function N times,and participants run a process similar to the slot reservation stage to obtain the tokens anonymously. On the other side, each participant generates one secret random number, and runs a blinding operation (equivalent to an encryption). In the last stage, the TR and the CA needs to encrypt and decrypt each incentive tag, blind and unblind sign all the tokens, and verify them. Encryption and decryption theoretically consume a lot of

resources, but it is necessary for privacy considerations. And because the fields that need to be encrypted and decrypted are short, the consumption in CA and TR is actually limited compared to the data transmission process; the subsequent experiments have also proved it.

6. Protocol Implementation

In order to prove the feasibility and efficiency of our protocol, we built and tested a prototype of the protocol. In this section, the implementation overview and the evaluation results will be illustrated and discussed.

6.1. Implementation Overview. The prototype uses java for both Android mobiles and servers. The involved cryptographic primitives are implemented using bouncy castles (sponge castles for Android). For the public key cryptosystem, all the server entities opt RSA-OAEP with a key length of 1024-bit and the participants use different keys.

For experiments, we ran the server on a PC (Intel XEON E3-1230, 3.3 GHz, 16 GB RAM, and windows 10) and an android phone (Google nexus 3). The network is 100 m WiFi channel for ensuring communication between the mobile phone and the server. The server runs three independent entities: SP, TR, and CA. For the N group members, first we run the setup stage to generate the full system parameters, and then we run N members on the same mobile phone where the execution is running in an independent fashion; therefore, we take only significant execution time. The communication between N members is simulated via 100 m WiFi channel where the server acts as a message repeater to send messages back to the phone.

The energy consumption of the phone battery is measured using an app named "iTest," which allows the extraction of energy consumption for each app on android 4.3. And, we use a formatted phone to reduce the error due to other app or system consumption. We also eliminate our own app's basic consumption (the energy consumption when the app is running without doing anything), which consumes a mean of 1.71193×10^{-2} J per second.

6.2. Performance Evaluation. The evaluation results of all stages are illustrated and discussed in this section. It is important to mention that privacy-related researches still lack metrics, which allow the comparison of different approaches. We can only compare similar approaches as comparing communication anonymity in our case where we can only compare protocols' latencies and energy consumptions.

6.3. Setup Stage. First, the setup stage is evaluated. Each member generates $N-1$ secret seeds of 32 bytes length. A secure pseudo random generation function based on SH1PRNG uses about 3.3×10^5 ns, and it uses about 25 ms to generate secrets for all the members of group size $N = 40$.

6.4. Slot Reservation Stage. Firstly, we evaluate the slot reservation stage latency of group size $N = 40$ for different numbers of slots, as shown in Figure 5. Because slot collisions may occur, the slot reservation possibly needs to repeat. For a certain number of slots, the bars illustrate the average time of submitting the slot reservation message once and finishing the slot reservation stage (i.e., all members hold a slot without any collision), respectively. Generally, to use more slots, it is necessary to not only bring a lower probability of collision, which reduces the risk to repeat to submit slot reservation message, but also increase the latency of pseudorandom generation and XOR operation. Therefore, the time of signal submission is rising with the growth of the number of slots, and the completion times are irregular. The ratio of completed time to single time shows the average counts of repeat. As expected, the average counts are 3 and 1.6 times for group size $M = 80$ and $M = 1600$, respectively. Nevertheless, the shortest completion time is 0.758 s when M = 200.Even though the single submission time is 0.361 s, which is longer than M = 80, the completion time is 0.0581 s faster.

As shown in Figure 6, the latency of the full slot reservation stage for different group sizes. For the OADR scheme, the number of slots is 5 times the group size (e.g., using 25 and 200 slots for 5 and 40 members, respectively). As shown, the increasing efficiency of the OADR scheme is dramatic. Even for a group size of 40, the average latency of the OADR scheme is less than 1 s (0.796 s). In the same condition, the latencies of Yao's and ADR scheme are 107.7 s and 26.8 s, respectively. It is worth noting that the error ranges of OADR scheme latency are relatively large. Among them, the longest experimental time is 1.152 s and the shortest is only 0.64 s in the case of 40 members. The reason is also obvious, the slot reservation stage needs to be rerun once the slot collision occurs in OADR scheme; this will not happen in other schemes. Also, the error ranges of other schemes' latencies depend only on the operations, while that of the OADR scheme is multiplied by the probability of slots collision occurring.

Next, we illustrate the traffic data overhead of slot reservation while the SRM length is 512 bits (64 bytes). As shown in Figure 7, for an SRM of 64-byte length, the traffic overheads are 435.156 KB and 165.761 KB for a group of 40 members in the Yao's and ADR schemes, respectively, while the traffic overhead of the same group under the OADR scheme increases to 509.4 KB. Due to the use of redundant slots, the proposed optimization increases the traffic overhead. But, the encryption operation is not used, the traffic overhead does not increase too much, and the runtime of the slot reservation is greatly reduced.

To measure the mean energy consumption of the one slot reservation stage and compare it to the other two slot reservation schemes, we run all schemes in similar conditions many times (10 times for each group size), and measure the total consumption. Since the app is the same, we do not have to substrate the base energy use. The mean energy consumption of three schemes for each participant for different group sizes is shown in Figure 8. Obviously, OADR scheme's consumption is largely lower than others to the same group size proportionally; As latencies, the error

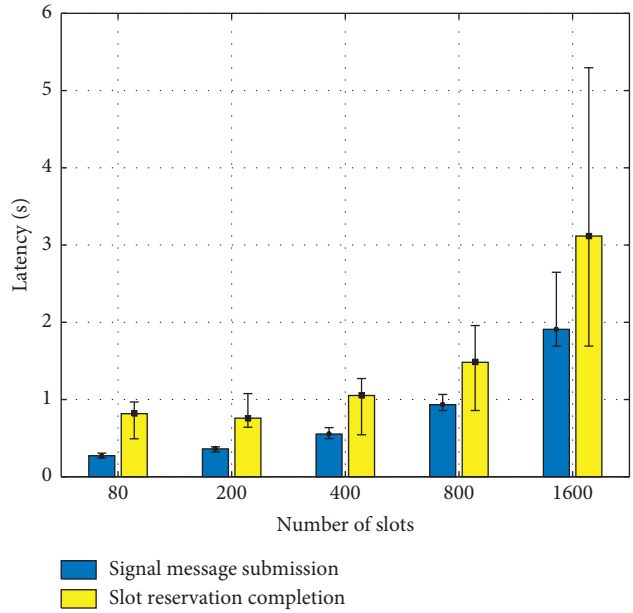


Figure 5: Slot reservation stage latency of group size $N = 40$ for different numbers of slots.

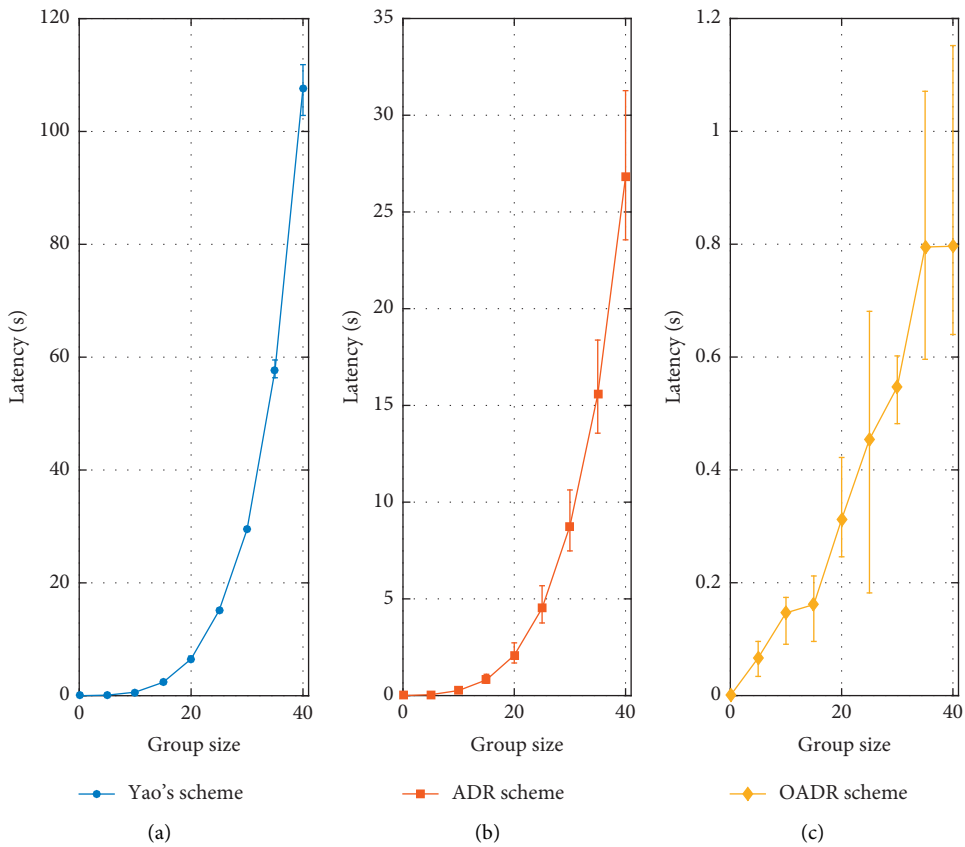


Figure 6: Latency of slot reservation stage for different group sizes.

ranges of OADR scheme consumption is still relatively large as the slot reservation stage needs to rerun if slot collision occurs. Nevertheless, it incurs 0.1537 J for a group of 40 participants, while the consumptions of ADR and Yao's scheme are about 3.7926 J and 0.8961 J, respectively, for the same group size.

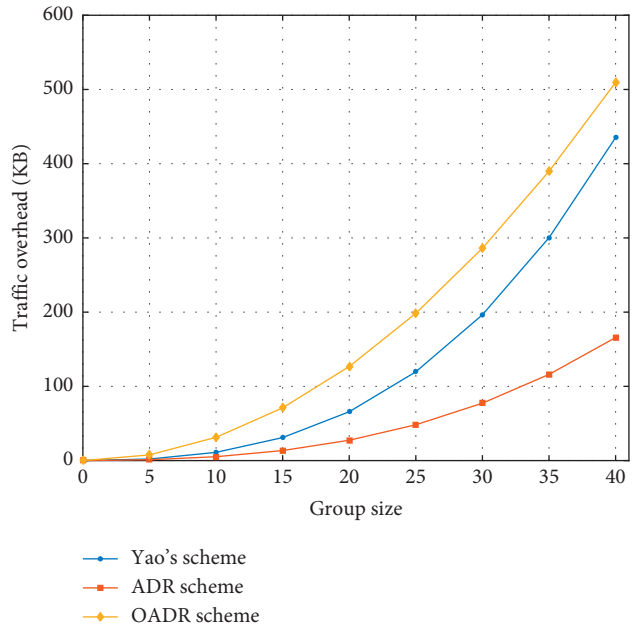


Figure 7: Traffic data overhead for slot reservation stage.

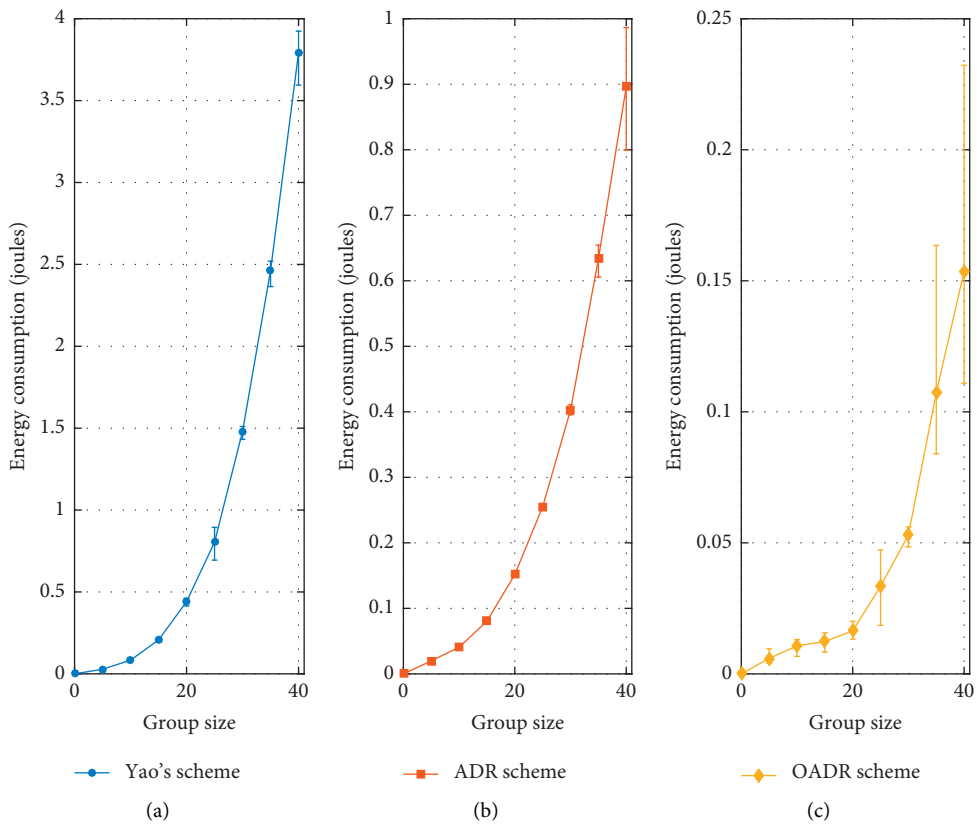


Figure 8: Consumption of slot reservation for different group sizes.

6.5. Data Submission Stage. We evaluate the data submission stage efficiency in terms of latency. For simplicity, we use the same data length for all members, and then we compute the latency of data submission for increasing group size starting from 0 to 40 with a 5-step increment. Because the ADR and Yao's schemes use the same data submission, only ADR and OADR schemes are compared here. Figure 9 shows the data submission latency for different schemes. The data lengths

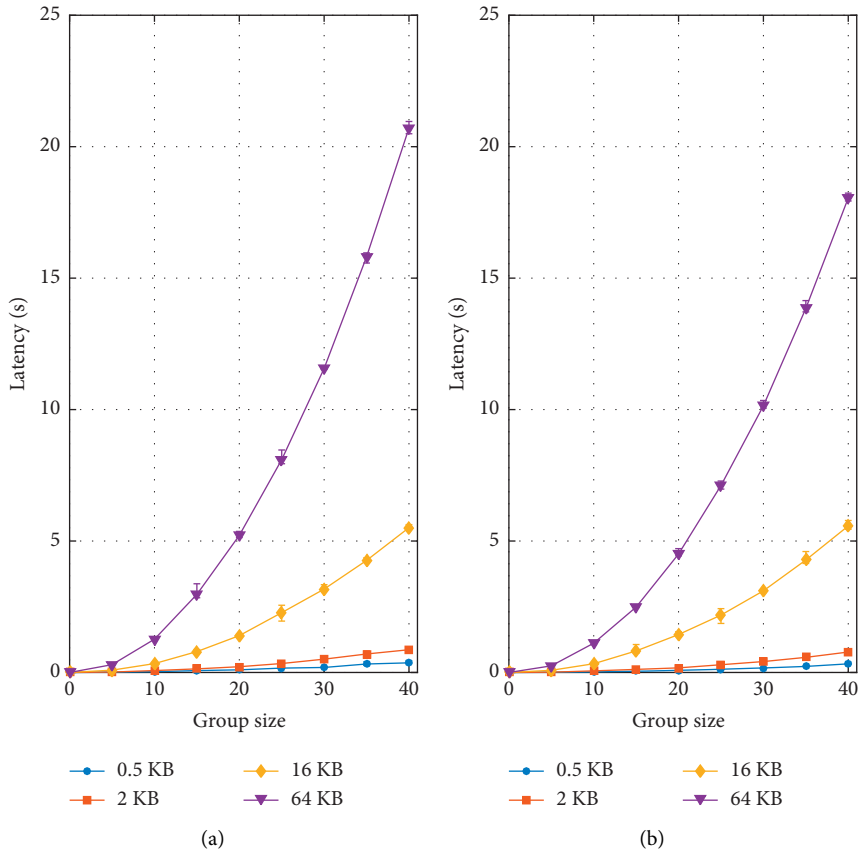


Figure 9: Latencies of data submission for different group sizes. (a) ADR scheme. (b) OADR scheme.

are 0.5 KB, 2 KB, 16 KB, and 64 KB, respectively. When the data lengths are short (0.5 KB and 2 KB), the latencies increase slowly as the group sizes increases, which for a 40-member group are 0.366 s and 0.865 s for 0.5 KB and 2 KB data lengths, respectively, using the ADR scheme. In the same condition, the OADR scheme uses 0.327 s and 0.777 s to submit the data. They are acceptable latencies for low-latency applications. For 64 KB of data length, the proportion of N Pseudo random and XOR function computing latency is significantly increased, and the overall latencies increase dramatically as the group increases. The ADR and OADR schemes grow to reach 20.71 s and 18.05 s, respectively, when considering a group of 40 members. The latency is reduced by approximately 10%. Although OADR does not reduce the number of bits in XOR operations, the number of generations of pseudorandom is reduced to $1/N$ (N is the group size) of the ADR scheme.

The energy consumption increases with the increase of the group size, although there are some fluctuations. The increase of the group size will increase the number of secret seeds and thus the number of calculations of the pseudo-random function. These fluctuations are also due to low consumption, which is easily affected by the particle of the energy monitoring of the Android system. On the other hand, the total stream bit vector length increases relatively to the group size. As shown in Figure 10, it is similar to the latency experiment; the consumptions increase slowly as the group sizes increase for the short data length. They are 1.0391 J and 0.8212 J at 0.5 KB data length for ADR and OADR scheme, respectively. For the larger data lengths, the consumption overhead begins to be large and noticeable. With 64 KB data, they are 36.6649 J and 32.1684 J, respectively. Compared to the ADR, the consumption reduction rate of the OADR scheme is more than 10%.

For participating sensing applications, the amount of data submitted is usually small. Therefore, bulk transfer does not lead to significant additional latency or energy consumption. When group size is 40 to transfer 2 KB data, the latency and energy consumption are still lower than 0.8 s and 1.6 J, respectively. On the other hand, the group size will also significantly affect the latency and energy consumption, so that the latency and energy consumption can be better restricted by reducing the group size. Therefore, the bulk transfer is suitable for low latency tolerant applications of participating sensing .

6.6. Token Request and Deposit Stages. We evaluate the incentive scheme by computing the latency of token signing by TR and token verification by CA for increment group size (Figure 11). The latency of blind signature signing for a group of 40 members is about 61.9 ms (a mean of 1.5 ms per member), which is negligible compared to data submission latency. For the verification of the same number of members, the consumed latency is about 73.5 ms, even it performs

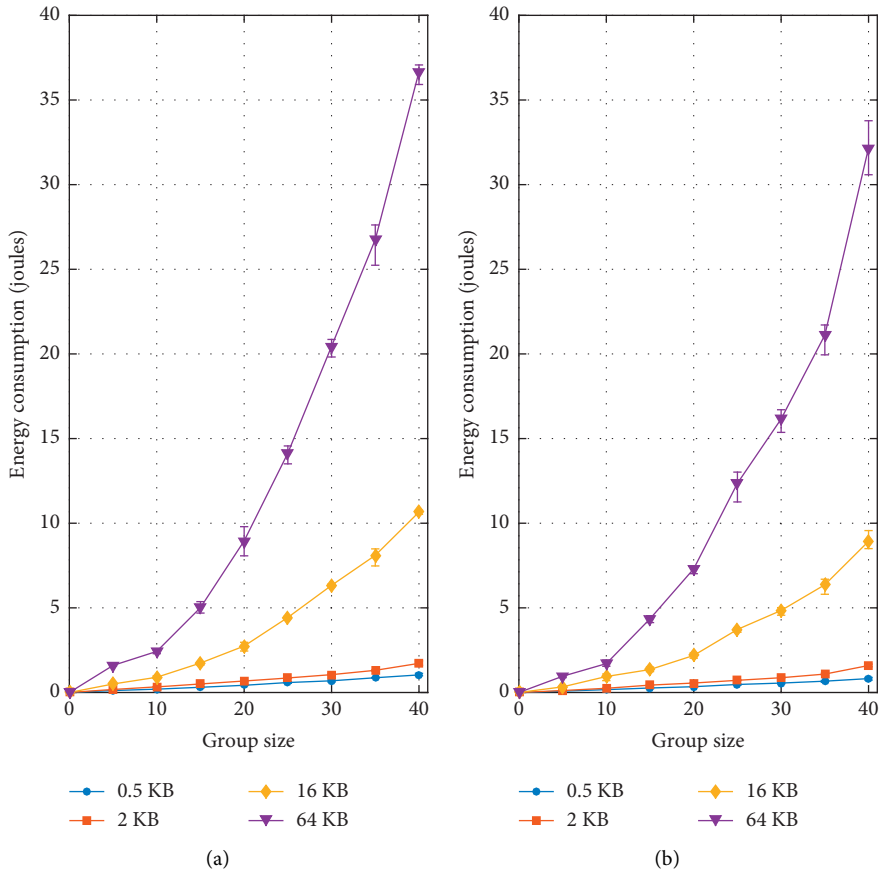


Figure 10: Consumption of data submission for different group sizes. (a) ADR scheme. (b) OADR scheme.

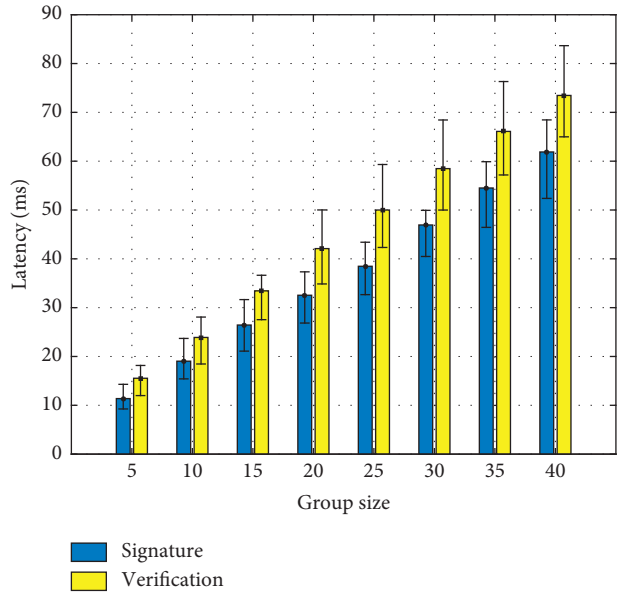


Figure 11: Latencies of blind signature in TR and verification in CA for different group sizes.

encryption, decryption, and an unblind signature. Thus, our proposed micropayment mechanism is efficient. In addition, Table 2 shows the time it takes for the mobile phone to generate and cancel the blind signature, where the process of generating the blind signature includes generating the r factor. The blind signature scheme does not consume a lot of

Table 2: Blind signature related time and consumption in mobile phone device.

Time and consumption	Interval	Average (ms)
Blinding signature time	[3.215 ms, 7.721 ms]	4.062
Blinding signature consumption	[0.052 J, 0.141 J]	0.081
Unblinding signature time	[1.831 ms, 4.6 ms]	2.21
Unblinding signature consumption	[0.024 J, 0.027 J]	0.025

resources, and the signature only spends 4.06 ms for one participant. Therefore, it fits perfectly to be used on phones.

7. Conclusions

In order to solve the privacy exposure by associating participant identities with multimodal information hidden or attached to reported data, we proposed an anonymous data reporting strategy with dynamic incentive mechanism for participatory sensing. The proposed protocol leverages the verifiable random slot selection to establish a transmission plan and then uses multiplayer DC-nets and bulk transfer for anonymous data submission, so that the participant's identity, which is linked to the multimodal information hidden in the data, is confused among a group of participants. The incentive mechanism uses a method similar to data transmission to break the connection between additional information and the participant's identity, and combines blindly signed tokens to anonymously verify the availability of the participant's identity. Furthermore, it reversely uses the blind signature mechanism to carry the dynamic incentives to anonymously complete micropayments transfer. This method is also suitable for the transfer of other similar additional information. In addition, we improve the bulk transfer used in previous works to increase the transmission efficiency while maintaining the same anonymity. The integrity, anonymity, and efficiency of this protocol are proved by theoretical analysis. We implemented and tested the prototype of our protocol. Experimental results on Android phones show that the protocol has advantages comparing with similar anonymous data reporting protocols. These results also show that this protocol is effective for low latency tolerance applications with a certain delay tolerance, which is the case for most participatory sensing applications.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by National Key R&D Program of China under Grant No. 2020YFB1710200, National Natural Science Foundation of China under Grant No. 62072236, and Fundamental Research Funds for the Central Universities under Grant No. 3072020CFT0603.

References

[1] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick, "A survey on privacy in mobile participatory sensing applications," *Journal of Systems and Software*, vol. 84, no. 11, pp. 1928–1946, 2011.

[2] Q. Li and G. Cao, "Providing efficient privacy-aware incentives for mobile sensing," *IEEE International Conference on Distributed Computing Systems*, vol. 15, no. 6, pp. 208–217, 2014.

[3] J. Son, D. Kim, R. Hussainy et al., "Privacy aware incentive mechanism to collect mobile data while preventing duplication," in *Proceedings of the IEEE MILCOM*, pp. 1242–1247, Tampa, FL, USA, October 2015.

[4] Q. Li and G. Cao, "Providing privacy-aware incentives in mobile sensing systems," *IEEE Transactions on Mobile Computing*, vol. 15, no. 6, pp. 1485–1498, 2016.

[5] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, pp. 1–12, 2018.

[6] X. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher, "Enabling reputation and trust in privacy-preserving mobile sensing," *IEEE Transactions on Mobile Computing*, vol. 13, no. 12, pp. 2777–2790, 2014.

[7] H. Wu, L. Wang, and G. Xue, "Privacy-preserving and trustworthy mobile sensing with fair incentives," in *Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC)*, IEEE, Shanghai, China, May 2019.

[8] Y. Li, Y. Zhao, S. Ishak et al., "An anonymous data reporting strategy with ensuring incentives for mobile crowd-sensing," *Journal of Ambient Intelligence & Humanized Computing*, vol. 9, no. B, pp. 1–15, 2017.

[9] J. Brickell and V. Shmatikov, "Efficient anonymity-preserving data collection," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 76–85, Philadelphia, PA, USA, August 2006.

[10] H. Corrigangibbs and B. Ford, "Accountable anonymous group messaging," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pp. 340–350, 2010.

[11] H. Corrigangibbs, D. I. Wolinsky, and B. Ford, "Proactively accountable anonymous messaging in verdict," in *Proceedings of the 22nd USENIX Conference on Security*, pp. 147–162, Washington, DC, USA, August 2013.

[12] D. Wolinsky, H. CorriganGibbs, and B. Ford, "Dissent in numbers: making strong anonymity scale," in *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation*, pp. 179–192, Hollywood, CA, USA, October 2012.

[13] X. Zhao, L. Li, G. Xue et al., "Efficient anonymous message submission," in *Proceedings of the IEEE INFOCOM 2012*, pp. 228–236, Orlando, Florida, USA, June 2012.

[14] Y. Yao, L. T. Yang, N. N. Xiong et al., "Anonymity-based privacy-preserving data reporting for participatory sensing," *IEEE Internet of Things Journal*, vol. 2, no. 5, pp. 381–390, 2015.

[15] D. Christin, "Privacy in mobile participatory sensing: current trends and future challenges," *Journal of Systems and Software*, vol. 116, pp. 57–68, 2016.

[16] D. Das, P. Mohan, and V. N. Padmanabhan, "Prism: platform for remote sensing using smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, pp. 63–76, San Francisco, CA, USA, June 2010.

[17] P. Sui and X. Li, "A privacy-preserving approach for multimodal transaction data integrated analysis," *Neurocomputing*, vol. 30, pp. 56–64, 2017.

[18] F. Zhang, H. Li, W. He et al., "Data perturbation with state-dependent noise for participatory sensing," in *Proceedings of the IEEE INFOCOM 2012*, pp. 2246–2254, Orlando, FL, USA, March 2012.

[19] F. David, F. Kargl, and L. Hans, "PUCA: a pseudonym scheme with strong privacy guarantees for vehicular ad-hoc networks," *Ad Hoc Networks*, 2016.

[20] S. Zhang, G. Wang, Q. Liu, X. Wen, and J. Liao, "A trajectory privacy-preserving scheme based on dual-k mechanism for continuous location-based services," in *Proceedings of the 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, pp. 1004–1010, IEEE, Guangzhou, China, December 2017.

[21] H. Zhao, Y. I. Xiao-Ling, and J. L. Wan, "Privacy-area aware all-dummy-based location privacy algorithms for location-based services," *DEStech Transactions on Computer Science and Engineering aice-ncs*, 2017.

[22] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, pp. 37–48, Baltimore, MD, USA, June 2005.

[23] C. Bettini and D. Riboni, "Privacy protection in pervasive systems: state of the art and technical challenges," *Pervasive and Mobile Computing*, vol. 17, pp. 159–174, 2015.

[24] Y. Wang, Z. Cai, Z. Chi, X. Tong, and L. Li, "A differentially *k*-anonymity-based location privacy-preserving for mobile crowdsourcing systems," *Procedia Computer Science*, vol. 129, pp. 28–34, 2018.

[25] F. Fei, S. Li, H. Dai, C. Hu, W. Dou, and Q. Ni, "A *k*-anonymity based schema for location privacy preservation," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 2, pp. 156–167, 2019.

[26] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS 1982)*, pp. 160–164, Washington, DC. USA, November 1982.

[27] H. Jiang and Q. Xu, "Advances in key techniques of practical secure multi-party computation," *Journal of Computer Research & Development*, vol. 52, no. 10, pp. 2247–2257, 2015.

[28] M. Blanton and E. Guiar, "Private and oblivious set and multiset operations," *International Journal of Information Security*, vol. 15, no. 4, pp. 493–518, 2016.

[29] J. Zhong, W. Wu, C. Cao et al., "A variable weight privacy-preserving algorithm for the mobile crowd sensing network," *Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 3053202, 7 pages, 2017.

[30] J. Xiong, R. Ma, L. Chen, Y. Tian, L. Lin, and B. Jin, "Achieving incentive, security, and scalable privacy protection in mobile crowdsensing services," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 8959635, 12 pages, 2018.

[31] J. Zhang, C. Zhe, L. Ting et al., "Design scheme of electronic jury system based on secure multi-party computation," *Journal of Computer Applications*, vol. 40, no. S2, pp. 80–84, 2020.

[32] J. Cheon, A. Kim, M. Kim et al., "Homomorphic encryption for arithmetic of approximate numbers," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 409–437, Hong Kong, China, December 2017.

[33] S. Fletcher and M. Z. Islam, "Differentially private random decision forests using smooth sensitivity," *Expert Systems with Applications*, vol. 78, pp. 16–31, 2016.

[34] M. Bun and T. Steinke, "Concentrated differential privacy: simplifications, extensions, and lower bounds," *Theory of Cryptography. TCC 2016*, https://arxiv.org/abs/1605.02065, 2016.

[35] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," *Journal of Machine Learning Research*, vol. 17, no. 17, pp. 1–51, 2016.

[36] N. Holohan, D. J. Leith, and O. Mason, "Extreme points of the local differential privacy polytope," *Linear Algebra & Its Applications*, vol. 534, pp. 78–96, 2017.

[37] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Minimax optimal procedures for locally private estimation," *Journal of the American Statistical Association*, vol. 113, no. 521, pp. 182–201, 2018.

[38] Q. Ye, H. Hu, X. Meng et al., "PrivKV: key-value data collection with local differential privacy," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP) IEEE*, pp. 127–143, San Francisco, CA, USA, May 2019.

[39] L. Sun, J. Zhao, X. Ye et al., "Conditional analysis for key-value data with local differential privacy," 2019, https://arxiv.org/abs/1907.05014v1.

[40] Y. Sei and A. Ohsuga, "Differentially private mobile crowd sensing considering sensing errors," *Sensors*, vol. 20, no. 10, p. 2785, 2020.

[41] T. Ni, Z. Chen, G. Xu et al., "Differentially private double auction with reliability-aware in mobile crowd sensing," *Ad Hoc Networks*, vol. 114, Article ID 102450, 2021.

[42] C. Cornelius, A. Kapadia, D. Kotz et al., "Anonysense: privacy-aware people-centric sensing," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, pp. 211–224, Breckenridge, CO, USA, June 2008.

[43] E. D. Cristofaro and C. Soriente, "Extended capabilities for a privacy-enhanced participatory sensing infrastructure (PEPSI)," *IEEE Transactions on Information Forensics and Security*, vol. 8, pp. 2021–2033, 2013.

[44] R. Shokri, G. Theodorakopoulos, P. Papadimitratos et al., "Hiding in the mobile crowd: locationprivacy through collaboration," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, pp. 266–279, 2014.

[45] F. Qiu, F. Wu, and G. Chen "Slicer, "A slicing-based k-anonymous privacy preserving scheme for participatory sensing," in *Proceedings of the IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems*, pp. 113–121, Hangzhou, China, October 2013.

[46] D. Chaum, "Blind signatures for untraceable payments," in *Proceedings of the CRYPTO'82*, pp. 23–25, Santa Barbara, CA, USA, August 1982.

[47] D. Chaum, "Blind signature system," in *Proceedings of CRYPTO'83, Advances in Cryptology*, pp. 21–24, Santa Barbara, CA, USA, August1983.

[48] S. Gisdakis, T. Giannetsos, and "S. Papadimitratos, "Security and privacy-preserving architecture for participatory-sensing applications," in *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 39–50, Oxford, UK, July 2014.

[49] Y. Qiu, M. Ma, S. Chen et al., "An anonymous authentication scheme for multi-domain machine-to-machine communication in cyber-physical systems," *Computer Networks*, vol. 129, 2017.

[50] T. Gao, Q. Wang, X. Wang, and X. Gong, "An anonymous access authentication scheme based on proxy ring signature for CPS-WMNs," *Mobile Information Systems*, vol. 2017, pp. 1–11, 2017.

[51] Y. Gong, C. Ying, Y. Guo et al., "A privacy-preserving scheme for incentive-based demand response in the smart grid," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1304–1313, 2017.

[52] X. Wang and D. Reeves, *Traceback and Anonymity*, Springer Publishing Company, Incorporated, New York, NY, USA, 2015.

[53] D. Christin, J. Guillemet, A. Reinhardt et al., "Privacy-preserving collaborative path hiding for participatory sensing applications," in *Proceedings of the IEEE Eighth International Conference on Mobile Ad Hoc and Sensor Systems*, pp. 341–350, Valencia, Spain, October 2011.

[54] S. Goldwasser and S. Micali, "Probabilistic encryption how to play mental poker keeping secret all partial information," in *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pp. 365–377, San Francisco, CA, USA, May 1982.

[55] B. Laurie, "Lucre: anonymous electronic tokens," 2008, https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=90F5A5D1F07340A314BB493534C50866?doi=10.1.1.112.7291\&rep=rep1\&type=pdf.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5586335, 12 pages
https://doi.org/10.1155/2021/5586335

Research Article

EX-Action: Automatically Extracting Threat Actions from Cyber Threat Intelligence Report Based on Multimodal Learning

Huixia Zhang,[1,2] **Guowei Shen**,[1,2] **Chun Guo**,[1,2] **Yunhe Cui**,[1,2] **and Chaohui Jiang**[1,2]

[1]*College of Computer Science and Technology, Guizhou University, Guiyang 550025, China*
[2]*Guizhou Provincial Key Laboratory of Public Big Data, Guiyang 550025, China*

Correspondence should be addressed to Guowei Shen; gwshen@gzu.edu.cn and Chun Guo; gc_gzedu@163.com

Received 4 February 2021; Accepted 7 May 2021; Published 27 May 2021

Academic Editor: Liguo Zhang

With the increasing complexity of network attacks, an active defense based on intelligence sharing becomes crucial. There is an important issue in intelligence analysis that automatically extracts threat actions from cyber threat intelligence (CTI) reports. To address this problem, we propose EX-Action, a framework for extracting threat actions from CTI reports. EX-Action finds threat actions by employing the natural language processing (NLP) technology and identifies actions by a multimodal learning algorithm. At the same time, a metric is used to evaluate the information completeness of the extracted action obtained by EX-Action. By the experiment on the CTI reports that consisted of sentences with complex structure, the experimental result indicates that EX-Action can achieve better performance than two state-of-the-art action extraction methods in terms of accuracy, recall, precision, and F1-score.

1. Introduction

With the increasing amount of information in modern society, advanced persistent threat (APT) attacks, as a new development of cyber security, have gradually become one of the main attack methods. APT attacks have many characteristics, such as long duration, complex attack methods, and strong concealment. The traditional defense method is a passive defense method, which mainly relies on security equipment and rules matching to generate an alarm for static control. It is not suitable for the protection of APT, 0 day attacks, and other new network security threats [1]. Therefore, many organizations have aimed to develop timely, relevant, and actionable CTI about emerging threats and key threat actors to enable effective cybersecurity decisions [2].

CTI is a kind of information that records current and former security threats [3], which contain information such as the reasoning, context mechanism, observable indicators, mitigation measures, and countermeasures of attacks. It is extremely time-consuming for security practitioners to analyze and utilize multisource and unstructured CTI reports. Therefore, automatic and efficient information extraction from unstructured CTI reports has become one of the main research directions.

Information extraction is the extraction of valuable information from unstructured CTI reports. The extracted information mainly includes cybersecurity entities and its relationship. Cybersecurity entity recognition identifies named entities in the cybersecurity field, which mainly include names of persons, organizations, places, and some security terms. Entity-relationship extraction is to extract the relationships between security entities in unstructured CTI reports. It is mainly for the triple extraction of known entities and predefined relationships. Complex relationships between entities with contextual connections are hard to be identified.

The action consists of the subject, the verb, and the object. Actions not only describe the attack behaviors in the attack process but also include non-predefined entities and their contextual semantic relationships. Therefore, actions are crucial for CTI reports. The subject and the object in actions correspond to a pair of security entities, and the verb describes the semantic relationship between the entity pair.

The entities and the relationship between them do not need to be predefined in the proposed method.

At present, the extraction of actions is mainly based on semantic dependency [4], and the ontology model [5] is used to identify them. Therefore, there are mainly the following challenges in the extraction and identification of threat actions:

(1) Threat actions cannot be accurately extracted in unstructured CTI reports just relying on their semantic dependency.

(2) Relying on ontology methods to identify actions will lose some undefined key threat actions.

(3) Information content of extracted threat actions is incomplete, and it is difficult to measure the information content of the extracted threat actions.

In this study, we propose a multimodal learning approach, named EX-Action, to accurately extract and automatically identify threat actions in unstructured CTI reports. EX-Action is a method based on the combination of mutual information and NLP technology. It can extract more actions based on the syntactic structure. Three main contributions of this study are listed as follows:

(1) We propose an actions extraction framework, named EX-Action. EX-Action extracts threat actions from the unstructured CTI reports that consisted of complex sentence structure by syntactic rule matching. And then, it identifies threat actions by a multimodal learning algorithm.

(2) We use an evaluation indicator named normalized mutual information (NMI) [6] to measure the difference of information content of threat actions, which quantifies the completeness of the information content of threat actions.

(3) We apply EX-Action to extract 18210 actions from 243 unstructured CTI reports, and the experimental result shows that the obtained accuracy, F1-score, and NMI of EX-Action are 79.09%, 85.58%, and 85.26%, respectively.

The rest of this study is organized as follows. We list the related work of extracting information from the CTI report in Section 2. In Section 3, we introduce the EX-Action framework and describe it. Section 4 gives the experimental results. In Section 5, we discuss the proposed method. Finally, Section 6 summarizes this study.

2. Related Work

The fragmentation of information in the era of big data gives unstructured CTI reports the characteristics of diversification, fragmentation, and heterogeneity. For these characteristics of unstructured CTI reports, Liao et al. [7] proposed an approach to automatically recover valuable attack indicators from popular technology blogs and convert them into industry-standard and machine-readable CTI reports. Sara Qamar et al. [8] proposed the construction of the Structured Threat Information eXpression (STIX) analyzer ontology and its ontology model relationship. Their method can determine the threat relevance, possibility, and affect and expose assets by automatically classifying network threats and formulated rules and inferences. Xun et al. [9] proposed an automatic identification model of threat intelligence (TI) based on a convolutional neural network (CNN) for automatically extracting TI from various unstructured TI data sources. These studies reduce the noise data in the CTI report by reorganizing unstructured threat report knowledge to identify cyber threat information in an effective manner.

It is one of the important research contents in CTI analysis that reconstruct CTI knowledge by using the graph mode. Shu et al. [10] used a graph model to organize multisource heterogeneous threat data, which formalize cyber threat intelligence computing into a new security paradigm. Ya et al. [11] proposed an attack entities recognition method to construct a CTI knowledge graph. Jia et al. [12] used existing machine learning technology to organize the knowledge of threat reports and construct a knowledge base of cybersecurity. Du et al. [13] proposed a knowledge graph for human-readable CTI recommendation from the perspective of the attack chain. The threat intelligence knowledge graph helps security practitioners understand cyber threats in a timely and rapid manner.

The current research on CTI reports mainly includes real-time perception, dynamic sharing, and effective application. Regarding the application of CTI, it contains structured and unstructured CTI reports. For structured CTI reports, Kim et al. [14] automatically generate rules without human intervention to mitigate new network security threats that have been discovered in real-time. In response to the lack of domain knowledge analysis under the existing structured CTI reports, Tappeiner et al. [15] proposed a domain recognizer based on a convolutional neural network to identify targeted domain of CTI and automatically generates specific CTI from social media data.

In order to solve the problem of overreliance on the analysis of security practitioners results in the inefficiency of CTI applications, Zhu et al. [16] proposed an end-to-end approach for automatic feature engineering, which identifies abstract behaviors that are associated with malware and map these behaviors to concrete features and generates a characteristic semantic network. Zhu et al. [17] proposed an approach to bridge measurement data with manual analysis and train a multiclass classifier to extract IOCs and further categorize them into different stages. Ayoade et al. [18] have leveraged natural language processing techniques to extract attacker's actions from threat report documents generated by different organizations and then automatically classify them into standardized tactics and techniques.

For threat actions extraction from unstructured CTI reports, Husari et al. [5] proposed a method named TTPDrill to extract actions based on semantic dependence and an ontology database, which is used to map actions to different attack patterns. However, TTPDrill will neglect part of threat actions in clause structure and parallel sentences. And it used ontology structure to identify threat actions, which will lose some undefined threat actions in the ontology structure.

Husari et al. [19] developed an approach named ActionMiner, which used NLP technology and based on information entropy and mutual information, to extract low-level cyber threat actions from publicly available CTI sources. However, ActionMiner has relied on syntactic analysis to extract low-level threat actions. It lacks a behavioral subject, and the information content is difficult to guarantee.

This study proposes a framework called EX-Action. It extracts actions based on the syntactic structure and rules mapping and identifies them by a multimodal learning algorithm. EX-Action identifies actions based on multiple features, which improves the accuracy of action recognition and covers actions in complex sentence structures.

3. Proposed Framework

In this study, we propose a framework called EX-Action. It contains four modules, which are data preprocessing, candidate threat actions extraction, action feature extraction, and action identification. The EX-Action architecture is shown in Figure 1. First, EX-Action preprocesses the obtained CTI report. Second, candidate threat actions are extracted by a rule-based method. And then, candidate action multimodal features are calculated. Finally, EX-Action identifies actions and generates selected actions by a weighted ensemble learning algorithm.

3.1. Data Preprocessing. In this module, EX-Action cleans the data of CTI reports by filtering invalid characters and sentences that do not contain threat actions. There are some cybersecurity terms in CTI reports. However, these cybersecurity terms are not recognized by NLP technology, such as file paths, IP addresses, and so on. EX-Action uses regular expressions to replace and save unrecognizable terms.

3.2. Candidate Threat Actions Extraction. In this module, EX-Action extracts candidate threat actions from preprocessed CTI reports by a rule-based method. A CTI report consists of n sentences, which can be expressed as $T = \{S_1, \ldots, S_i, \ldots, S_n\}$, and each sentence contains several action verbs, $S_n = \{V_1, \ldots, V_i, \ldots, V_N\}$. For each verb, EX-Action extracts many actions based on a rules-matching strategy, denoted as $A_i = \{\text{action}_1, \text{action}_2, \ldots, \text{action}_m\}$. The extracted candidate threat actions are in the format (subject, verb, and object), i.e., an action consists of three elements which are subject, verb, and object. EX-Action matches the parts of speech (POS) for the three elements that consist of the action. POS are used to match the elements in action. The three elements in action rule matching are given in Table 1. The column "POS" represents the POS of each component, and "POS-Symbols" represent the symbol of POS tagged.

In this module, sentences are tagged by the POS tool [20]. Take the verb that is identified in the results of POS tagging as the start of the sliding window. Then, the subject and object are, respectively, searched in the threat description sentence. The window size for searching the subject and object can affect its extraction performance. Some potential objects may have a long distance from the target verb, and therefore, a too small window size cannot get them. However, a too large window size may result in many mismatched VO pairs, which will affect the identification efficiency of EX-Action. EX-Action adopts different strategies to set the sliding window size for searching the subject and the object. For the subject, the sliding window size is set to the number of words before the verb in one sentence, and then, EX-Action matches all nouns and noun combinations in the window with the verb. For the object, a dynamic window mechanism is used to set the sliding window size. This mechanism adopts the number of words after the verb in one sentence as the sliding window size, and the sliding window stops sliding when encountering another verb. Figure 2 shows an example of the action extraction of EX-Action.

In the process of searching the subject and object, there are phenomena that a noun compound structure or pronouns act as the subject or object. To ensure the integrity of the extracted action information, the multinoun compound structure is tokenized as a noun and matched with a verb. The verb and object can retain the basic information content, but when the pronoun is acted as the object, a lot of information might be lost. Therefore, the pronoun as the subject is saved, and the pronoun as the object is discarded in this module.

3.3. Action Feature Extraction. In this module, EX-Action extracts five types of features for each action. The extraction framework of action's features is shown in Figure 3. It contains similarity measurement, probability computation, mutual information value measurement, semantic dependency measurement, and distance computation. The features contains 9 values, $\text{feature}_{\text{action-all}} = \{F_1, F_2, \ldots, F_9\}$. The description of features is given in Table 2. More details of action feature extraction will be described next.

3.3.1. Similarity Measurement. In this subsection, the similarity between candidate actions and a CTI report p is calculated by the TF-IDF and BM25 algorithms. The TF-IDF method is commonly used to calculate the feature item weight in the process of text vectorization [21]. Equation (1) is used to calculate the weight of a feature item of the action.

$$\text{TF} - \text{IDF}(x) = \frac{N}{N(x)} * \left(\log\left(\frac{N+1}{N(x)+1}\right) + 1\right), \quad (1)$$

where N is the total number of words in the CTI report p, and $N(x)$ represents the number of words x in the CTI report p. Since threat actions contain different numbers of words, the average value is used as the similarity measure of candidate actions.

The BM25 [22] is an upgraded algorithm of TF-IDF. It adds a constant to TF-IDF to limit the growth limit of the TF value and uses the document length to evaluate the

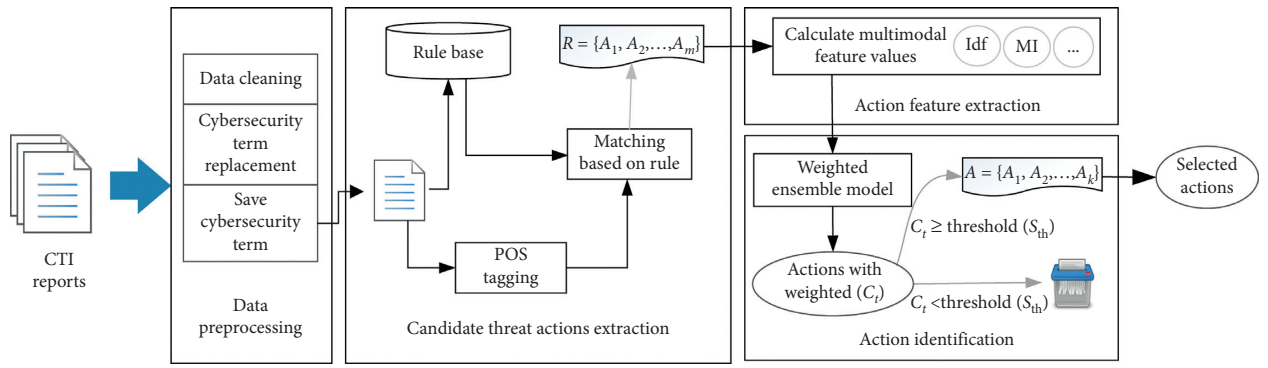


Figure 1: The architecture of EX-Action.

Table 1: The three elements in action rule matching table.

Element in action	POS	POS-symbols
Subject	Noun, noun and noun, pronoun, noun + cardinal number	NN, NNS, NNP, NNPS, PRP, CC, N + CD
Verb	Verb, verb and verb	VB, VBD, VBG, VBP, VBN, VBZ
Object	Noun, noun and noun, noun + cardinal number	NN, NNS, NNP, NNPS, CC, N + CD

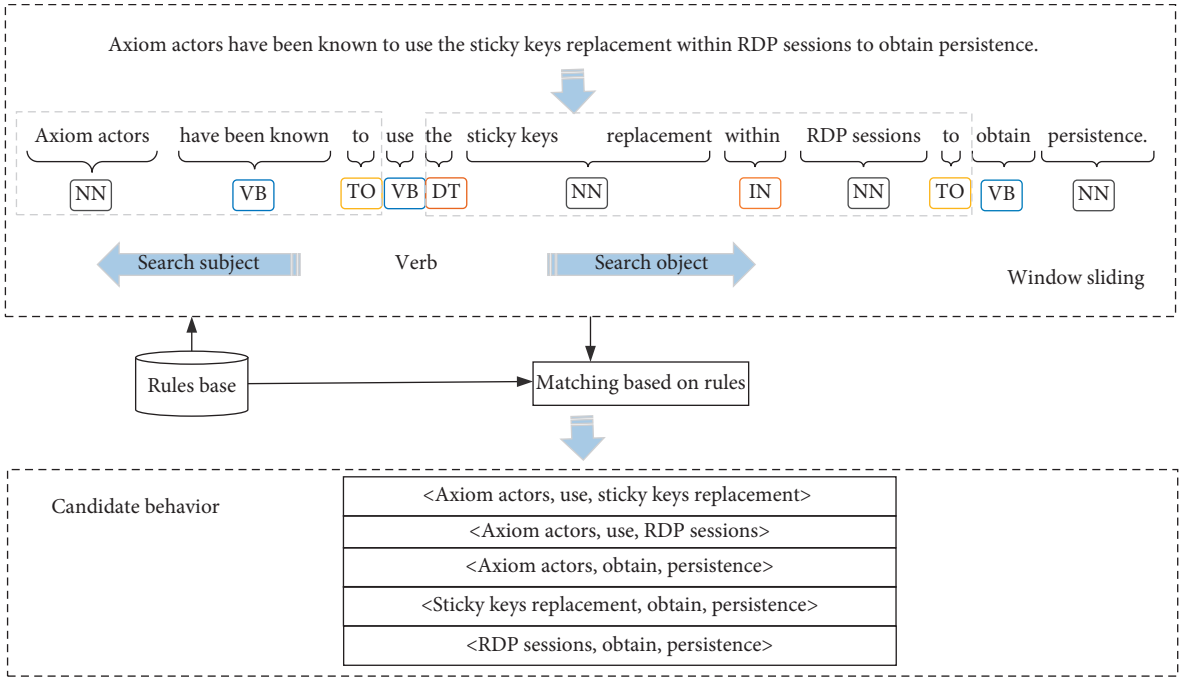


Figure 2: An example of threat action extraction.

importance of candidate actions. It performs a weighted summation on the correlation scores between candidate threat actions and the CTI report p, and equation (2) is used to calculate the BM25 of the action.

$$\text{Simlarity} = \text{idf} * \frac{((k_1 + 1) * \text{tf})}{(k_2 * (1 - b + b * L) + \text{tf})}, \quad (2)$$

where tf represents the frequency of each word, idf represents the inverse word frequency of each word, L is the length of the text, and k_1, k_2, and b are the adjustment factors.

3.3.2. Probability Computation. In this subsection, the co-occurrence frequency of the VO pair is calculated to determine the correlation between the candidate action and the CTI report. In action, the subject usually indicates the attacking subject or organization, the verb represents the attack action, and the object represents the operation target in the CTI report. Since attack organizations are different in the attack process, calculating the co-occurrence frequency of SVO triples will weaken the relevance between the action and the CTI report. Therefore, EX-Action calculates the co-occurrence frequency of the VO pair under a fixed window is taken as a feature. The correlation between the action and the

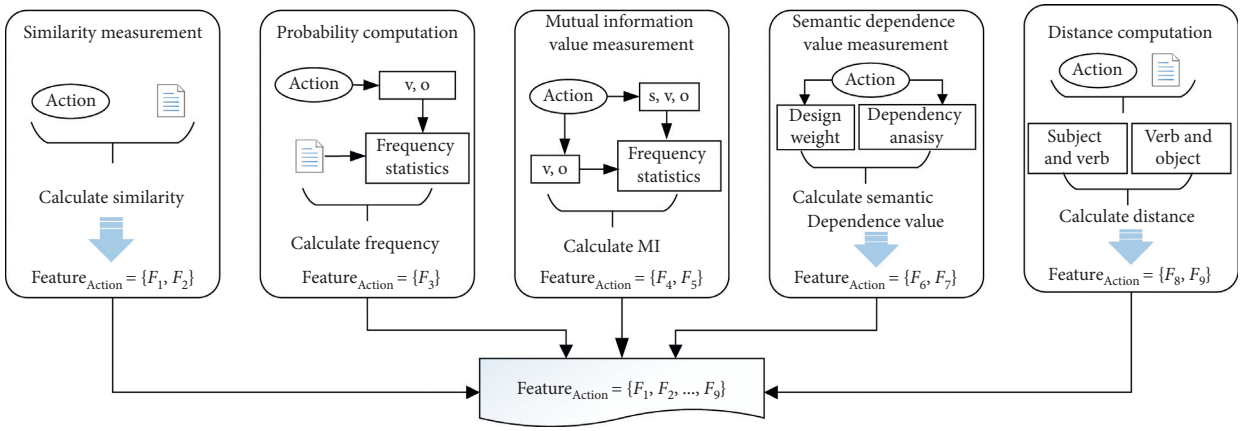


Figure 3: The extraction framework of action feature.

Table 2: The description of five type features.

Type	Feature	Description
Similarity	TF − IDF (F_1), BM25 (F_2)	The similarity between the action and description sentences.
Probability	P_vo (F_3), frequency (F_4)	The probability and frequency of the VO pair.
Mutual information	MI_VO (F_5), MI_SVO (F_6)	The MI contains two values, which, respectively, are the VO pair-MI (MI_VO) and the SVO triples-MI (MI_SVO).
Semantic dependence	Dependence (F_7)	The semantic dependency of each word in action.
Distance	Distance_SV (F_8), distance_VO (F_9)	It contains two values, which, respectively, express the distance between the verb and the subject (distance_SV) and the distance between the verb and the object (distance_VO).

CTI report is proportional to the frequency of the VO pair. Specifically, window size $m = 25$ is used to calculate the co-occurrence frequency of the VO pair in our experiment.

3.3.3. Mutual Information Value Measurement. Mutual information (MI) measures the reduction in uncertainty of information about one random variable, given knowledge of another [23]. The MI of VO pair and SVO triple are calculated to measure the information content of candidate actions. The correlation between candidate actions and CTI report is proportional to the MI value. Equation (3) is used to calculate the MI of SVO triple.

$$\mathrm{MI} = \sum_{s \in S} \sum_{vo \in VO} \log \frac{p(s, \mathrm{vo})}{p(s) * p(\mathrm{vo})}, \tag{3}$$

where $p(s, \mathrm{vo})$ is the frequency of a threat action, $p(s)$ is the number of occurrences of its subject, and $p(\mathrm{vo})$ is the co-occurrence frequency of VO pair.

3.3.4. Semantic Dependence Measurement. There are some candidate actions with high matching degree, but they in fact are inaccurate semantic matching. The feature of semantic dependency is designed to recognizing these actions. The Stanford dependency analyzer [4] is used to analyze the relation of semantic dependency for each sentence. W_1 and W_2 are set as the dependency weight between the subject and the verb and the dependency weight between the verb and the object, respectively. And then, summing the dependency weight (W_1 and W_2) as the feature of semantic dependency, Figure 4 shows an example of Stanford semantic dependency for a sentence.

3.3.5. Distance Computation. In this subsection, two distances are computed. They are, respectively, the distance between subject and verb and the distance between verb and object. The number of word between the verb and the target word is taken as the value of distance. For example, for a word between the subject and the verb, the distance is recorded as 1.

3.4. Action Identification. Ensemble learning promote weak learners to strong learners by constructing and combining multiple base learners to complete the learning task. In this module, the EX-Action automatically identifies candidate actions by a parallel ensemble learning algorithm. The main process is illustrated in Algorithm 1. The time complexity of Algorithm 1 is $O(n^2)$.

In Algorithm 1, the training set D is the input, which contains candidate actions and their features. The ground truth A is the action set of manual extraction. The ground truth A is used to calculate the similarity of candidate actions. Five-base classification learners are used to construct a parallel ensemble classification. They are, respectively, decision tree (tree), random forest (forest), support vector machine (SVM), linear regression (LR), and multilayer perceptron classifier (MLPC). It can be expressed as $T = \{T1, T2, \ldots, T5\}$.

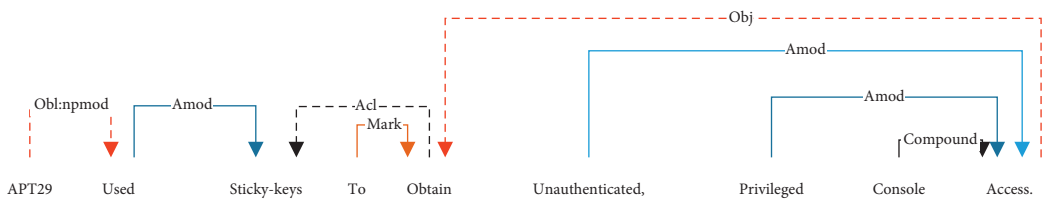


Figure 4: An example of Stanford semantic dependency for a sentence.

Require:
Input:
Training Set: $D = \{(\text{Action}_1^M, \text{Feature}_1), (\text{Action}_2^M, \text{Feature}_2), \ldots, (\text{Action}_m^M, \text{Feature}_m)\}$; //$\text{Action}^M$ is the candidate action of machine identification, Feature_n is the $n-th$ action feature value, which contains 9 values
$A = \{\text{Action}_1^H, \text{Action}_2^H, \ldots, \text{Action}_k^H\}$; //$\text{Action}^H$ is the action of manual extraction
Base classification learners: $T = \{T_1, T_2, \ldots, T_5\}$
Train:
for each $t \in [0, m]$**do**
  for each $i \in [0, 5]$**do**
    $\text{result}_i = T_i(\text{Action}_t^M, \text{Feature}_t)$//$\text{result}_i$ is the predicted value of action in the $i-th$ Base classification Learners
  end for
  $C_t = \sum_{i-1}^{m} \text{result}_i * \omega_i // \omega_i$ is the weighted of action in the $i-th$ Base classification Learners
  if $C_t > S_{th} // C_t$ is the $t-th$ action weight, S_{th} is the optimal voting threshold **then**
    Threat action$\leftarrow \text{Action}_t^M$//Threat action is the selected action set
    $S_t = f(\text{Action}_t^M, \text{Action}_t^H) // S_t$ is the similarity of the $t-th$ action between the ground truth and machine identification
  end if
  if $S_t > \theta // \theta$ is the optimal similarity threshold **then**
    Action_t^M is correct action
  end if
end for
Output: Threat Action $= \{\text{Action}_1^M, \text{Action}_2^M, \ldots, \text{Action}_k^M\}$

Algorithm 1: Threat actions identification.

The result_i is the predicted value for the action generated by the i-$^{\text{th}}$ base classification learner. Then, different weight ω_i is set for the result_i and sum them to gain the weighted C_t of each action. EX-Action identifies selected actions from candidate threat actions set by the weighted voting method and minimizes the loss function through the linear combination of base learners. S_{th} is a predefined voting threshold; if C_t is larger than S_{th}, the candidate action will be regarded as the selected action. Finally, EX-Action calculates the similarity S_t between the selected action and the ground truth. If the similarity S_t is larger than the predefined similarity threshold θ, the action Action_t^M is recognized as the correct action.

In EX-Action, according to the different classification performances of the different models, different weight values are set for each model. It can be seen that the decision tree behaves the best performance in our experiment. Therefore, the decision tree is assigned to the maximum weight, and the weight values of the other four models are all equal.

4. Evaluation

4.1. Experimental Dataset. We obtained 243 security reports from ATT&CK[1]. They contain 5136 sentences. The number of sentences regarding different techniques in CTI reports is given in Table 3. In our experiment, 20% of the CTI reports are randomly chosen as the test data. Figure 5(a) shows the distribution of sentence lengths, and Figure 5(b) shows frequency distribution of the test data.

It can be seen from Figure 5 that the length of the sentence that describing threat action is mainly distributed between 10 and 30. Therefore, the dataset in this study can be regarded as the CTI report with complex sentence and long length.

4.2. Evaluation Metrics. In this study, accuracy, recall, precision, F1-score, normalized mutual information (NMI), and number of extracted actions (number) are used as performance metrics. The accuracy, recall, precision, and F1-score reflect the quantitative difference of threat actions between machine identification and the ground truth. They can be calculated by equations (4)–(7).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{6}$$

Table 3: The number of sentences regarding to different techniques in CTI reports.

Techniques	CTI	Sentences
Initial access	11	166
Execution	35	737
Persistence	38	376
Privilege escalation	17	165
Defense evasion	36	780
Credential access	16	264
Discovery	22	1048
Lateral movement	14	308
Collection	13	386
Command and control	20	654
Exfiltration	6	58
Impact	15	194
Total	243	5136

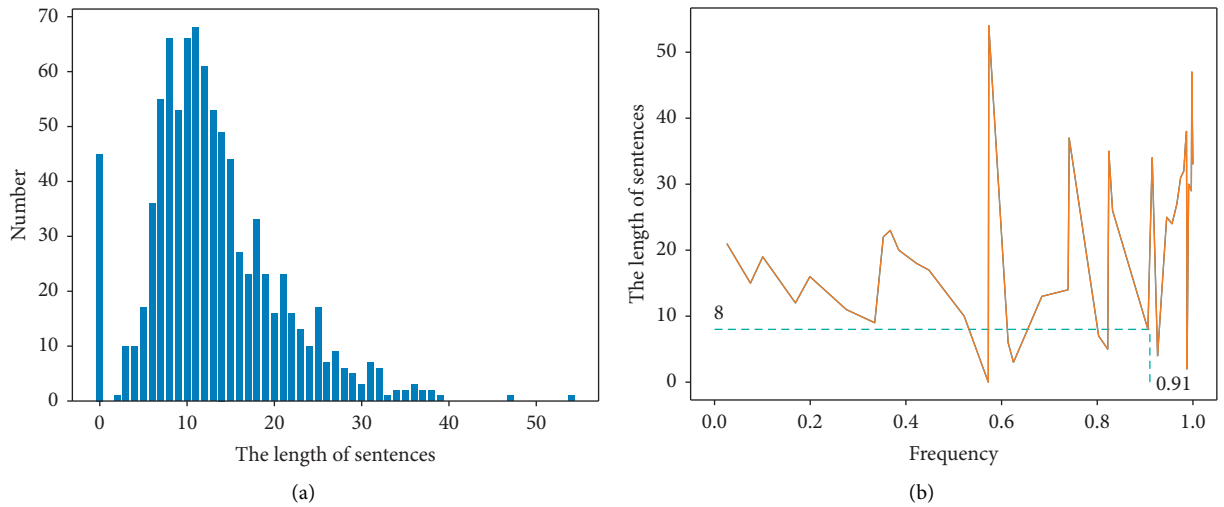


Figure 5: The number of sentence lengths and frequency distribution graphs of the test data. (a) The number of sentence lengths distribution. (b) The frequency distribution graphs of the test data.

$$\text{F1} - \text{score} = \frac{2 * \text{precision} * \text{recall}}{\text{Precision} + \text{recall}}. \tag{7}$$

The number represents the number of extracted actions, and NMI represents a measure of the difference in threat actions information content between machine recognition and manual extraction. NMI is often used in clustering to measure the similarity of two clustering results. And it is used to measure the difference in the information content in this study. The NMI reflects the similarity of actions between machine recognition and manual extraction. Equation (8) is used to calculate the information content difference of each action.

$$\text{NMI} = \frac{-2\sum_{i=1}^{C_m}\sum_{j=1}^{C_k} C_{ij} * \log C_{ij} * N/C_i * C_j}{\sum_{i=1}^{C_m} C_i * \log C_i/N + \sum_{j=1}^{C_k} C_j * \log C_j/N}, \tag{8}$$

where N represents the number of word nodes of the action, C_m represents the number of word nodes of the machine recognition action, C_h represents the number of word nodes of the manual extraction action, and C_{ij} represents the number of word nodes belonging to both types of actions. The similarity of information between the machine recognition and the ground truth is proportional to the MI value. When the NMI is equal to 1, the information content is equal.

4.3. Results and Analysis. In this section, four subsections are there to show our experimental results and analysis. They are the feature importance ranking of EX-Action, model comparison, threshold determination, and the effect comparison of existing methods. Note that the best values of each metric are bold in each table.

4.3.1. Feature Importance Ranking of EX-Action. This subsection shows the feature distribution of actions, the importance distribution of features, and the performance of different features combination. The features contain 9 values. They are $\text{TF} - \text{IDF}(F_1)$, $\text{BM25}(F_2)$, $P_{\text{vo}}(F_3)$, $\text{frequency}(F_4)$, $\text{MI}_{\text{VO}}(F_5)$, $\text{MI}_{\text{SVO}}(F_6)$, $\text{dependence}(F_7)$, $\text{distance}_{\text{SV}}(F_8)$, and $\text{distance}_{\text{VO}}(F_9)$. The feature distributions of some actions are shown in Figure 6. It can be seen that the feature value distributions of these actions are nonlinear distributions.

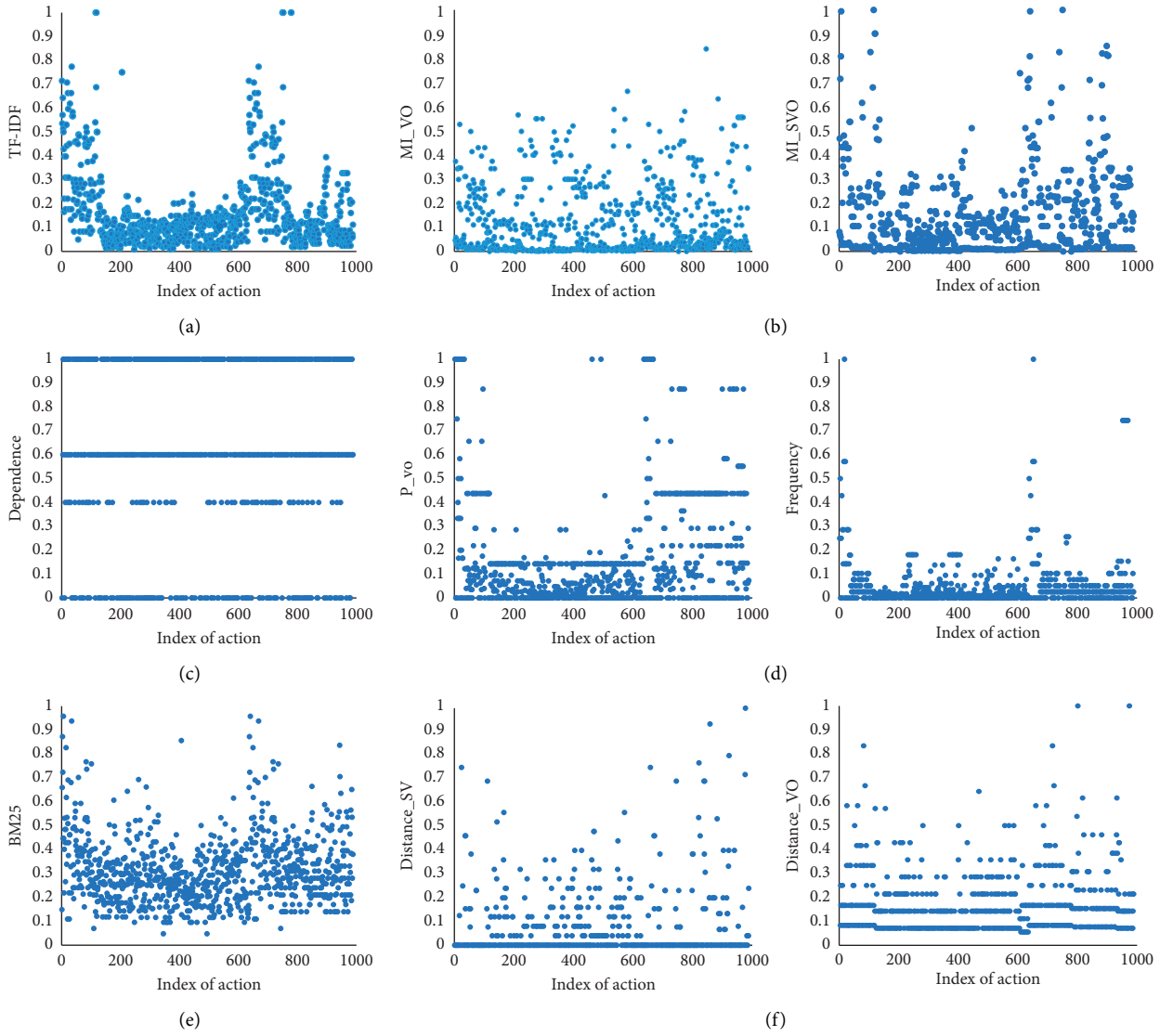


Figure 6: The different feature values distribution of threat actions. (a) Threat actions distribution of feature TF-IDF. (b) Threat actions distribution of feature MI. (c) Threat actions distribution of feature semantic dependence. (d) Threat actions distribution of feature VO pair frequency. (e) Threat actions distribution of feature BM25. (f) Threat actions distribution of feature window length.

Table 4 gives the obtained result of different features combinations. Under the same conditions, the recall of combination$_1$ and combination$_2$ reached the maximum value of 77.82%, and the number of extracted actions reached the maximum value of 1179, but other metrics are lower than combination$_7$. The performance of combination$_7$ is higher than others combinations in terms of accuracy, precision, F1-score, and information completeness. It can be found that combination$_7$ is more appropriate for the feature selection of threat action identification.

The importance distribution of the 9 features is calculated by the Gini index, as given in Figure 7. Figure 7 provides data that the distance of VO pairs has the largest effect on actions recognition. The frequency and the conditional probability of VO pairs are less important than other features.

4.3.2. Model Comparison. This subsection shows the results of the different base learners, unweighted ensemble learning model (unweighted model), and EX-Action (weighted ensemble model). The results obtained by the different base learners, unweighted model, and EX-Action are given in Table 5.

As given in Table 5, the accuracy and F1-score of tree are higher than other base learners, but its accuracy and F1-score are lower than EX-Action. Therefore, in EX-Action, the weight of the tree is the largest, and the weight values weights of other base learners are the same. Comparing the results of the unweighted model and EX-Action, the recall of the unweighted model is 81.06%, which is higher than EX-Action, and the number of extracted actions is also higher than EX-Action. However, the accuracy, precision, F1-score, and NMI values of EX-Action are higher than those of the unweighted model.

Table 4: Comparison of the result of different features.

Combination	Features	Number	Accuracy (%)	Recall (%)	Precision (%)	F1-score (%)	NMI (%)
Combination_1	$(F_1, F_2, F_5, F_6, F_7, F_8, F_9)$	1179	71.99	**77.82**	82.16	79.93	94.08
Combination_2	$(F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9)$	1179	71.29	**77.82**	80.98	79.36	94.02
Combination_3	$(F_1, F_3, F_4, F_5, F_6, F_7, F_8, F_9)$	1177	72.20	77.68	82.65	80.09	93.81
Combination_4	$(F_1, F_2, F_3, F_4, F_5, F_6, F_8, F_9)$	1169	71.28	77.16	81.63	79.33	93.98
Combination_5	$(F_1, F_2, F_3, F_4, F_5, F_6, F_7)$	1046	64.88	69.04	78.71	73.56	93.09
Combination_6	$(F_1, F_2, F_3, F_4, F_7, F_8, F_9)$	1131	69.82	74.65	81.72	78.03	93.76
Combination_7	$(F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9)$	1167	**72.35**	77.03	**83.60**	**80.18**	**94.26**

The bold values represent the maximum values of each metric.

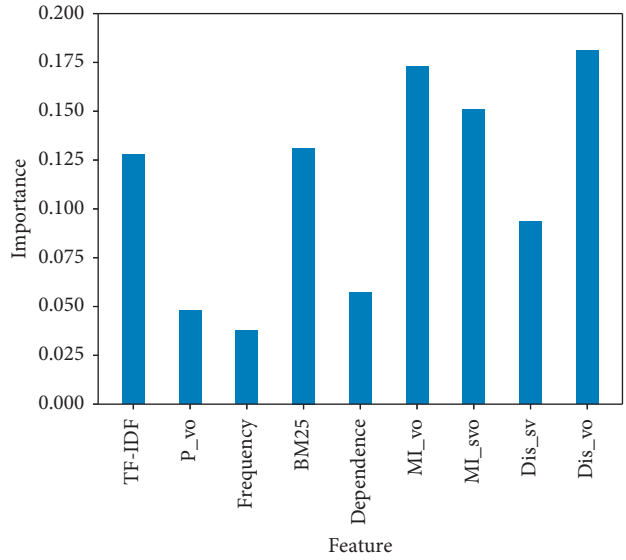


Figure 7: The importance distribution of features.

Table 5: Comparison of the result of different models.

Model	Number	Accuracy (%)	Recall (%)	Precision (%)	F1-score (%)	NMI (%)
SVM	726	52.25	47.92	79.43	59.78	93.14
Tree	1135	71.64	74.92	84.70	79.50	93.85
LR	736	50.90	48.58	73.75	58.58	93.33
MLPC	716	50.43	47.26	74.74	57.91	93.16
Forest	612	49.38	40.40	**88.95**	55.56	93.64
Unweighted model	1228	67.66	**81.06**	72.58	76.58	93.61
EX-Action	1167	**72.35**	77.03	83.60	**80.18**	**94.26**

The bold values represent the maximum values of each metric.

4.3.3. Threshold Determination. The voting threshold determines the result of the model recognition actions, and the similarity threshold determines the correctness of actions recognization, which will influence the performance of EX-Action. This subsection tests the optimal parameters for EX-Action through the setting of the voting threshold and similarity threshold. The comparison of results under different voting thresholds and different similarity thresholds is shown in Figure 8.

As shown in Figure 8(a), when the similarity threshold and voting threshold are set to 0.2 and 4, respectively, accuracy, F1-score, and NMI are optimal. Besides, as shown in Figure 8(b), the similarity threshold is the degree of difference between machine recognition action and ground truth. It can be seen that the higher the similarity threshold, the higher the information content it contains and the lower the accuracy and F1-score will be.

4.3.4. Performance Comparison with the Existing Approaches. In this subsection, the performance of EX-Action is compared with TTPDrill [5] and ActionMiner [19] in terms of accuracy, recall, precision, F1-score, the number of extracted actions, and NMI. As shown in Table 6, the result of EX-Action is higher than the other two methods in terms of accuracy, recall, precision, F1-score, the number of extracted actions, and NMI.

For extracting actions from the CTI reports with complex structures, TTPDrill mainly relies on semantic

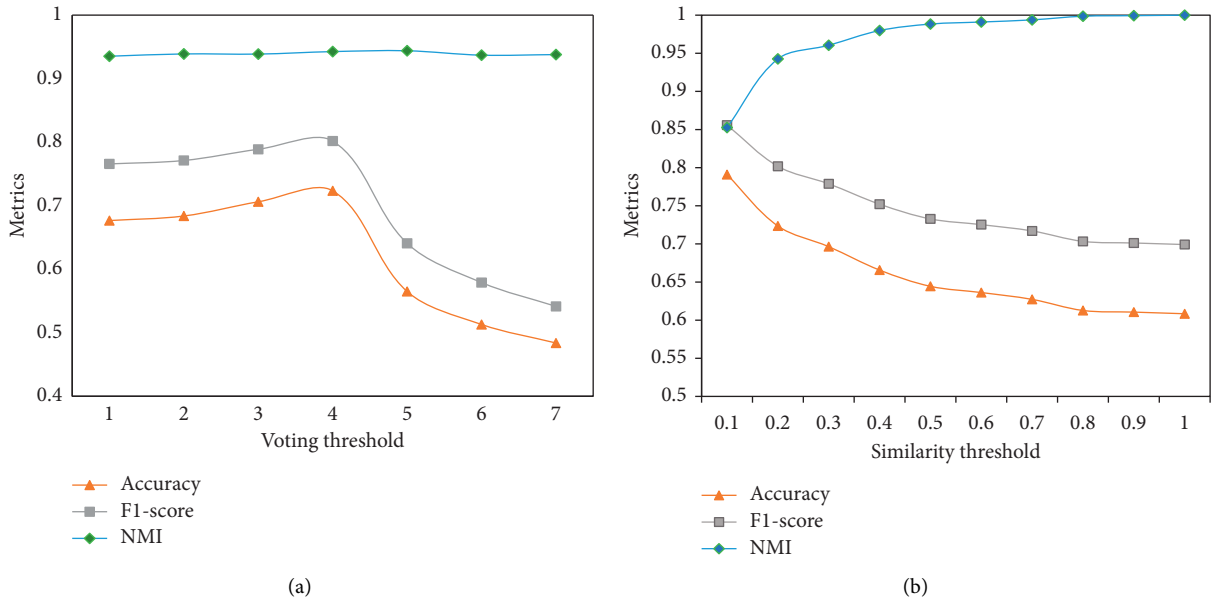


Figure 8: The effect of thresholds. (a) The effect of different voting thresholds. (b) The effect of different similarity thresholds.

Table 6: Performance comparison of the three methods.

Model	Number	Accuracy (%)	Recall (%)	Precision (%)	F1-score (%)	NMI (%)
TTPDrill	711	54.00	46.93	88.32	61.29	72.86
ActionMiner	1237	75.02	81.65	83.30	82.47	60.90
EX-Action	1246	**79.09**	**82.24**	**89.19**	**85.58**	**85.26**

The bold values represent the maximum values of each metric.

dependence. It will ignore part of threat actions in the complex sentence structure like clauses. Therefore, TTPDrill extracted fewer actions and behaved in poor performance. TTPDrill can retain the main information of the action compared with the ActionMiner, so the NMI is higher than ActionMiner. ActionMiner mainly relies on syntactic structure extraction. It can obtain better accuracy and recalls for low-level actions extraction for complex sentences. However, it does not retain the subject of the action, so its NMI value is low.

Besides, we compare the examples of actions extracted from CTI reports used in our experimental dataset and the literature [19] that proposed ActionMiner, respectively. The examples of the extracted actions obtained by the three methods in the two datasets are given in Table 7.

The actions extracted by the three methods on our experimental dataset are shown on the left of Table 7. It can be seen that TTPDrill has a better effect for extracting sentences with simple structure and obvious dependencies. Therefore, its performance is poor in our experimental dataset. ActionMiner can extract more actions than TTPDrill, but it lacks the subject of the action and behaves in poor performance in retaining the information content of the sentence. EX-Action can achieve better results in the number of extracted actions and the retention of the information of the sentences with complex structures.

For the CTI report mentioned in the literature [19], the actions extracted by the three methods are shown on the right of Table 7. It can be seen that the threat description sentence structure in those CTI reports is relatively simple. Comparing the three methods, it is found that actions extracted by TTPDrill can give a good description, but the composite components of the sentences are still not extracted. ActionMiner can extract more actions, but it lacks the subject. The actions extracted by EX-Action are more complete than that of ActionMiner, and the number of extracted actions extracted by EX-Action is more than TTPDrill.

5. Discussion

The unstructured CTI report records the network attack process, context mechanism, and other information. Accurately extracting and identifying threat actions from unstructured CTI reports will help security practitioners efficiently restore the attack process. In [24], Gao et al. correlated the threat action extracted from the CTI text with the action extracted from the system audit log and constructed a threat action graph to realize an efficient network threat search.

5.1. Contributions. First, EX-Action can be used to extract actions from unstructured CTI reports with complex sentences. It uses syntactic rules to extract threat actions, which can extract more actions in complex sentences. At the same time, machine learning algorithms are used to identify

Table 7: Threat actions extracted from CTI report in different datasets.

	The CTI report in our dataset	The CTI report mentioned in paper named ActionMiner
Threat description	APT29 used sticky keys to obtain unauthenticated, privileged console access. APT3 replaces the sticky keys binary executable file for persistence. Axiom actors have been known to use the sticky keys replacement within RDP sessions to obtain persistence. Deep Panda has used the sticky keys technique to bypass the RDP login screen on remote systems during intrusions. Empire can leverage WMI debugging to remotely replace binaries like executable file, executable file, and executable file with executable file.	It creates the following file: caches_version.db. . .. The Trojan creates the following registry entries. . .. Next, the Trojan steals the following information from the compromised computer: keystrokes, clipboard data, screenshot based on specified keywords in the window title, network adapter information such as MAC address, IP address, adapter name, adapter, and description. The Trojan then saves the stolen information in the following location: caches_version.db.
TTPDrill	APT29 used keys, Deep Panda used technique	Creates file: caches_version.db, Trojan creates registry entries, Trojan steals information from computer, Trojan saves information in location: caches_version.db. . .
ActionMiner	Used sticky keys, obtain console access, use sticky keys, use replacement, obtain persistence, has used technique, bypass RDP login, bypass screen, replace binaries	Creates file, creates registry entries, steals information, steals keystrokes, steals clipboard data, steals screenshot, steals network adapter information, steals MAC address, steals IP address, steals adapter name, steals adapter description, saves information
EX-Action	APT29 used sticky keys, APT29 obtain console access, axiom actors use sticky keys replacement, axiom actors obtain persistence, Deep Panda has used technique, Deep Panda bypass RDP login screen, empire replace binaries	It creates file caches_version, Trojan creates registry entries, Trojan steal information, Trojan steals computer keystrokes, clipboard data, screenshot, Trojan steals MAC address, IP address, adapter name, and adapter description, Trojan saves information, Trojan save location: caches_version

actions based on their own features, which can identify more actions undefined in the ontology model. Second, EX-Action extracted the action that contains the subject, verb, and object. It also provides a method to extract entity relations, which have contextual semantic relations between entities.

5.2. Limitations. There are some shortcomings in this study, such as overreliance on part-of-speech and semantic analysis, which may lose part of threat actions and failed to recognize the pronoun referent.

6. Conclusion

This study proposes a multimodal learning approach to extract and identify threat actions, and it can extract the threat action in complex semantic relationships and recognition of the cybersecurity entity associations with undefined relationships. The experimental result shows that EX-Action can have a certain balance between accuracy and information completeness in the action extraction. In future work, we will research how to avoid overreliance on part-of-speech tagging tools and try to use pronoun resolution to identify the subject and object of the pronoun.

Data Availability

The unstructured CTI reports data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (62062022) and the Science and Technology Foundation of Guizhou Province ([2017]1051).

References

[1] J. H. Li, "Overview of the technologies of threat intelligence sensing, sharing and analysis in cyber space," *Chinese Journal of Netword and Information Security*, vol. 2, no. 2, p. 16, 2016.

[2] S. Samtani, M. Abate, V. Benjamin, and W. Li, "Cybersecurity as an industry: a cyber threat intelligence perspective," *The Palgrave Handbook of International Cybercrime and Cyberdeviance*, vol. 2020, pp. 135–154, 2020.

[3] Y. Lin, P. Liu, H. Wang, W. J. Wang, and Y. Q. Zhang, "Overview of threat intelligence sharing and exchange in cybersecurity," *Journal of Computer Research and Development*, vol. 57, no. 10, p. 2052, 2020.

[4] D. Marneffe, M. Catherine, and C. D. Manning, "The Stanford typed dependencies representation," in *Proceedings of the Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*, pp. 1–8, Manchester, UK, August 2008.

[5] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, X. Niu, and " Ttpdrill, "Automatic and accurate extraction of threat actions from unstructured text of cti sources," in *Proceedings of the 33rd Annual Computer Security Applications Conference*, pp. 103–115, Orlando, FL, USA, December 2017.

[6] S. Lee, Y. T. Park, and B. J. d'Auriol, "A novel feature selection method based on normalized mutual information," *Applied Intelligence*, vol. 37, no. 1, pp. 100–120, 2012.

[7] X. J. Liao, K. Yuan, X. F. Wang, Z. Li, L. Y. Xing, and R. Beyah, "Acing the IOC game: toward automatic discovery and analysis of open-source cyber threat intelligence," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer*

and Communications Security, pp. 755–766, Vienna, Austria, October 2016.

[8] S. Qamar, Z. Anwar, M. A. Rahman, E. Al-Shaer, and B.-T. Chu, "Data-driven analytics for cyber-threat intelligence and information sharing," *Computers & Security*, vol. 67, pp. 35–58, 2017.

[9] S. Xun, X. Y. Li, and Y. L. Gao, "AITI: An automatic identification model of threat intelligence based on convolutional neural network," in *Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence*, pp. 20–24, Xiamen China, May 2020.

[10] X. K. Shu, F. Araujo, D. L. Schales et al., "Threat intelligence computing," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1883–1898, Toronto, ON, Canada, October 2018.

[11] Y. Qin, G.-W. Shen, W.-B. Zhao, Y.-P. Chen, M. Yu, and X. Jin, "A network security entity recognition method based on feature template and CNN-BiLSTM-CRF," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 6, pp. 872–884, 2019.

[12] Y. Jia, Y. Qi, H. Shang, R. Jiang, and A. Li, "A practical approach to constructing a knowledge graph for cybersecurity," *Engineering*, vol. 4, no. 1, pp. 53–60, 2018.

[13] M. Du, J. Jiang, Z. Jiang, Z. Lu, and X. Du, "PRTIRG: a knowledge graph for people-readable threat intelligence recommendation, Knowledge Science, Engineering and Management," in *Proceedings of the International Conference on Knowledge Science, Engineering and Management*, pp. 47–59, Springer, Athens, Greece, August 2019.

[14] E. Kim, K. Kim, D. Shin, B. Jin, and H. Kim, "CYTIME: Cyber Threat Intelligence ManagEment framework for automatically generating security rules," in *Proceedings of the 13th International Conference on Future Internet Technologies*, pp. 1–5, Seoul, Republic of Korea, June 2018.

[15] E. Tappeiner, F. Finotello, P. Charoentong et al., "TIminer: NGS data mining pipeline for cancer immunology and immunotherapy," *Bioinformatics*, vol. 33, no. 19, pp. 3140-3141, 2017.

[16] Z. Y. Zhu and T. Dumitras, "Featuresmith: automatically engineering features for malware detection by mining the security literature," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 767–778, Vienna, Austria, October 2016.

[17] Z. Zhu and T. Dumitras, "Chainsmith: automatically learning the semantics of malicious campaigns by mining threat intelligence reports," in *Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 458–472, IEEE, London, UK, April 2018.

[18] G. Ayoade, S. Chandra, L. Khan, K. Hamlen, and B. Thuraisingham, "Automated threat report classification over multi-source data," in *Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pp. 236–245, IEEE, Philadelphia, PA, USA, October 2018.

[19] G. Husari, X. Niu, B. Chu, and E. Al-Shaer, "Using entropy and mutual information to extract threat actions from cyber threat intelligence," in *Proceedings of the 2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pp. 1–6, IEEE, Vancouver, BC, Canada, May 2018.

[20] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60, Berlin, Germany, August 2014.

[21] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the First Instructional Conference on Machine Learning*, vol. 242, pp. 29–48, Citeseer, Washington, DC, USA, August 2003.

[22] M. Wijewickrema, V. Petras, and N. Dias, "Selecting a text similarity measure for a content-based recommender system," *The Electronic Library*, vol. 37, 2019.

[23] P. Latham and Y. Roudi, "Mutual information," *Scholarpedia*, vol. 4, no. 1, p. 1658, 2009.

[24] P. Gao, F. Shao, X. Y. Liu et al., "Enabling efficient cyber threat hunting with cyber threat intelligence," 2020, https://arxiv.org/abs/2010.13637.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6697862, 15 pages
https://doi.org/10.1155/2021/6697862

Research Article

Diffusion Analysis and Incentive Method for Mobile Crowdsensing User Based on Knowledge Graph Reasoning

Jian Wang, Shanshan Cui, Guosheng Zhao, and Zhongnan Zhao

School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China

Correspondence should be addressed to Jian Wang; wangjianlydia@163.com

Received 31 December 2020; Revised 9 February 2021; Accepted 18 March 2021; Published 8 May 2021

Academic Editor: Liguo Zhang

Aiming at the problem that the mobile crowdsensing (MCS) system relies on a specific platform with a large user group presupposed, this paper proposes a sensing user diffusion analysis and incentive method based on knowledge graph reasoning. We consider motivating users to participate under the constraint of limited budget so that the platform and users can get the most benefits. In this paper, we focus on socially aware users represented by self-organizing social networks, combine the knowledge graph to establish a knowledge graph for the crowdsensing system, use rules to derive user influence, and optimize user contributions. With the goal of maximizing social welfare, we propose a social awareness reverse auction (SARA) mechanism, in which the total contribution of users is the key to select winners, and the winners are paid based on critical prices. Through experimental simulations, we verify that SARA is close to the optimal social welfare under budget constraints.

1. Introduction

In recent years, mobile devices with various embedded sensors (such as smart phones and smart watches) have been everywhere, and vehicle-mounted and portable sensors (such as compass, accelerometer, GSP, and camera) have also appeared one after another. Mobile users carry their devices for extensive use in daily life, facilitating the information generation process. With the rapid development of various mobile sensing technologies, the human-centered mobile crowdsensing (MCS) system [1–3] has been developed. The mobile crowdsensing system has many applications in our daily life, covering many aspects, including medical care, environmental monitoring, intelligent transportation, and noise monitoring.

Participating in crowdsensing tasks is an expensive process for users. On the one hand, it will consume the user's resources, such as computing power, storage memory, and battery. On the other hand, this process may require the user to submit personal sensitive information, thereby causing the user's privacy to be leaked. Therefore, if there is no satisfactory reward to compensate the user's participation fee, the user will not be willing to participate in the perception task. However, the crowdsensing system depends on the total user participation level and the individual contribution of each user. Therefore, in order to stimulate and recruit users who use mobile sensing devices to participate in sensing tasks, it is important and challenging to design an incentive mechanism to achieve sustainable profitability of service providers.

There have been many studies on the incentive mechanism of mobile crowdsensing, but most of them assume that there are enough candidate users. However, this assumption may not be true in the real world. Especially mobile users may not even know it when the mobile crowdsensing app is just released to the public. In this case, the assumed large user pool no longer exists. Even if all candidate users are selected, the abovementioned worker selection method cannot achieve the high data quality of the mobile crowdsensing task. In this case, encouraging users to participate in sensing tasks cannot achieve good performance. Suppose that, after user u_i bids, he shares the sensing task T published by the platform with his friends and relatives through social networks. Among them, user u_j who is interested in the sensing task and participates in the bid is a member of u_i's recruitment target N_i. In this way, the task is

spread out through u_i, thereby expanding the set of sensing users. However, users will not spread freely. By analyzing the relationship between user u_i and the recruited users, the social utility generated by user u_i is obtained. Integrate user u_i's personal contribution and its social utility to make a winning bid selection, and calculate the reward of u_i based on the user's total contribution to complete the positive incentive.

The popularity of mobile social networks (MSN) (such as Facebook, Twitter, and Foursquare) has created a new medium for information sharing and dissemination. Pilot studies on real-world datasets [4–6] prove that it is feasible to promote novel products or innovative ideas by considering the interdependent behaviors of mobile users from the social field and the influence propagation on social networks [7, 8]. Traditionally, the network effect means that if a user purchases a public product or service, the more relevant users are willing to accept the purchase of a public product or service. In the mobile crowdsensing service, the participation behavior of mobile users can be regarded as purchasing ""public goods,"" which means that when users participate in sensing tasks, their related mobile users are more willing to participate, that is, users are easily influenced by acquaintances. Therefore, complex and interdependent user behaviors pose a major challenge to the operation of the crowdsensing platform. More importantly, network effects often exist in closely connected social relationships.

As shown in Figure 1, this paper attempts to recruit workers for MCS tasks in a novel way, using social networks as the recruitment platform and not relying on a specific MCS platform. First, we build a knowledge graph oriented to crowdsensing on the internal relationship of mobile users on MSN and conduct rule-based attribute inferences. Assuming the user pushes the task to their friends and the affected user accepts the task, the user can get a certain amount of extra rewards in return. The platform integrates user contributions and social effects to select winner users to maximize social welfare. Finally, the platform gives corresponding rewards, which are affected by the user's own social effects and contributions. The main contributions of this article are as follows:

(i) As far as we know, the introduction of knowledge graphs into the crowdsensing incentive mechanism is the first work. We build a knowledge graph of users' social relationships and rule to deduce the social influence between users, so as to obtain the social effects of users.

(ii) Incorporate the user's social network into the design of the incentive mechanism of mobile crowdsensing to encourage users to share recommendation behaviors, thereby promoting sensing tasks and obtaining more sensing users.

(iii) Design social awareness reverse auction (SARA) incentive method, select users based on the net contribution margin to reduce perceived data redundancy, and, at the same time, introduce users' social effects to optimize social welfare.

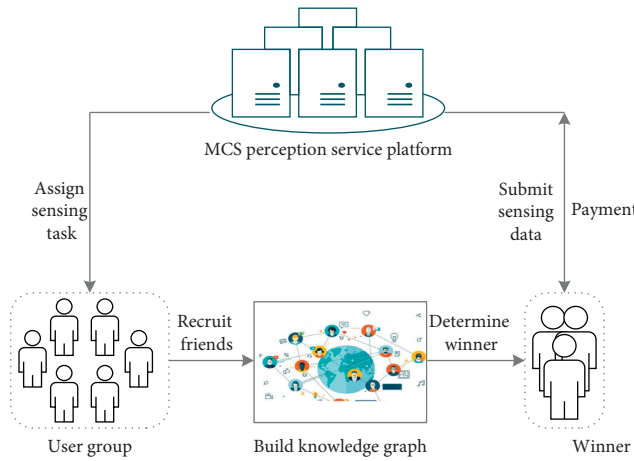


Figure 1: Mobile crowdsensing system structure.

(iv) Through experimental verification, SARA satisfies user rationality, budget feasibility, and computational efficiency. And, it is better than the comparison algorithm and close to the optimal social welfare.

2. Related Work

With the continuous development of the application of the crowdsensing system, a series of incentive mechanisms have been proposed [9], mainly from two aspects. On the one hand, it aims to maximize the sensing quality of crowdsensing (such as coverage) under certain restricted conditions. Literature [10] proposed an incentive mechanism considering budget constraints. The maximum weighted coverage under different coverage requirements in crowdsensing is modeled as a reverse auction with limited budget, which is used to maximize the weighted coverage in mobile crowdsourcing. Zhou et al. [11] proposed an online mechanism design to motivate users to participate and assign location-aware tasks to dynamic arrival users subject to capacity constraints (restrictions on the degree of participation). Construct a natural incentive mechanism through multiple rounds of online auctions in the desired time domain. Peng et al. [12] incorporated the consideration of data quality into the design of crowdsensing incentive mechanism. The rewards of participants are determined according to the quality and contribution of users, so as to encourage rational participants to effectively perform crowdsensing tasks. Jiang et al. [13] proposed to introduce a new data layer between the sensing task and the user. Using the similarity of perception tasks and the heterogeneity of users, a joint task selection and user scheduling problem is established on the data layer to maximize the social welfare of the entire system. Jin et al. [14] proposed an incentive framework for Thanos based on reverse combinatorial auctions. Combining the key indicator of worker's quality of information (QoI), authors design user selection and pricing algorithms for single task and multitask to maximize social welfare.

On the other hand, it aims to minimize the cost to ensure a certain level of sensing data quality. Jin et al. [15] designed an incentive mechanism based on reverse combinatorial auctions to achieve low-cost completion of relatively high-quality perception tasks while meeting certain quality requirements for perception tasks. Xu et al. [16] designed two incentive mechanisms MCT-M and MCTTS bidding models. The neural network method is introduced to learn the similarity between users, and the clustering algorithm is used to cluster users without their knowledge. Assign tasks based on the similarity between users to ensure the quality of perceived data. Duan et al. [17] proposed a reward-based collaboration mechanism. If a sufficient number of participants are willing to collaborate, CSP assigns tasks to collaborators and shares the total reward.

A common feature of the above mechanisms is that they are all proposed under the assumption that the number of participating users is sufficient. This is the essential difference from the mechanism mentioned in this paper. Some existing mechanisms propose to design incentive mechanisms based on social networks. Nie et al. [18] proposed to use a two-stage Stackelberg game to analyze the participation of mobile users and used reverse induction to analyze the optimal incentive mechanism of crowdsensing service providers. Zhao et al. [19] proposed a three-stage method for social sensing data distribution through D2D communication. This method selects initial seeds by fusing social networks and mobile networks and performs subsequent data forwarding by adapting to the user's altruistic and selfish incentive constraints, while ensuring the authenticity of the user. Sun et al. [20] considered the social relationship of SU and designed a social awareness incentive mechanism (SAIM). However, it only selects users based on their bids and does not consider the amount of data contribution that users can provide. In the above research, the dissemination of crowdsensing tasks among friends was not introduced into the recruitment of crowdsensing workers. Long et al. [21] proposed a community discovery algorithm based on multidimensional social relationship characteristics. By calculating the minimum spanning tree, connection parameters, and community integration among mobile nodes, the behavior patterns of nodes are abstracted and identified. The matching degree between the sensing task and the characteristic value of the community behavior pattern is calculated, and the distribution of the task is completed by the central node of the community according to the matching degree. Wang et al. [22] proposed that a part of seed users should be selected by considering users' social relationships and the differences (similarity) of trajectory prediction under the condition of constraining the number of seeds and the total number of workers. However, Wang et al. [22] did not consider the user's sensing quality into the incentive mechanism of crowdsensing and only analyzed the spread of users under social consciousness to select seed users. Although the method proposed in this paper also involves the influence propagation on social networks, it is different from the above research in the following aspects: (1) different optimization goals and constraints; (2) different worker selection algorithms; (3) different user influence reasoning models.

In this paper, various types of relevant information are extracted from social networks to assist MCS in inferring expertise. We optimize user contributions based on the user's preference for different tasks and comprehensively consider the social influence between users. The winning bidder is selected based on the user's contribution, and the user's critical reward is paid within the budget constraint.

3. User Social Relationship Reasoning

3.1. Construction of User Diffusion Based on Knowledge Graph. The knowledge graph oriented to crowdsensing is a network of relationships between users, tasks, geographic locations, and corresponding attributes. Users, tasks, and geographic locations as entities under the crowdsensing system are the most basic elements in the knowledge graph. The relationships describe the objective relationships between concepts, entities, and events, which exist between different entities. The entity relationships under this system include selection, recruitment, and location. The attribute refers to the specific attribute value under the entity, where user attributes include task tendency, common address, sensing time, and user bid price, and attributes of sensing task include task content, description, location, and time.

Figure 2 describes the user diffusion model of crowdsensing based on the knowledge graph. The model mainly includes three parts: data processing, the construction of ontology related to crowdsensing, and the generation of knowledge graphs of the crowdsensing system. The data source used is structured data, and the data is processed into a triple form to construct a resource description framework. At the same time, a top-down approach is used to construct ontology, which lays the foundation for information extraction. According to the resource description framework and ontology, entities related to crowdsensing and the relationships between entities are extracted. The model is a five-element crowdsensing knowledge base model that includes concepts, examples, relationships, attributes, and rules. The architecture of crowdsensing knowledge base contains three kinds of ontology: mobile node, task, and geographic location. The three entity types of ontology are mobile node, task, and geographic location. The mobile node in this paper is a node that carries sensors and can participate in sensing tasks, and users are a subclass of mobile nodes. The task is the sensing task issued by the platform, which is completed by the sensing user. Geographical location is a specific location where users perform tasks. The introduction and construction process of the basic elements of the crowdsensing user diffusion model based on the knowledge graph is mentioned above.

3.2. Relationship Inference Based on the Knowledge Graph. In the crowdsensing system, the user forwards the task and invites friends to participate in the sensing task. After the friend joins, it brings additional benefits to the user.

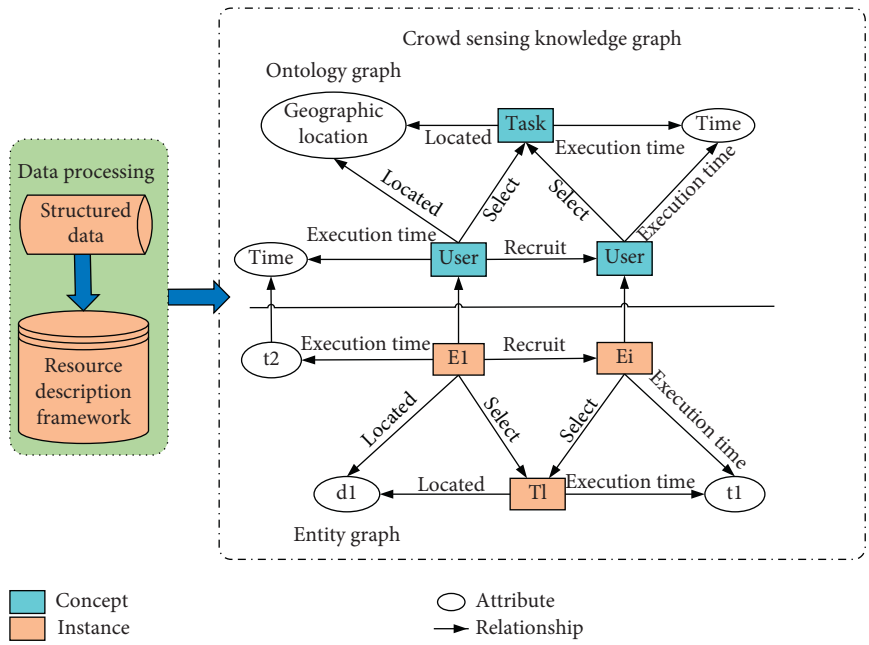


Figure 2: Crowdsensing user diffusion model based on knowledge graph.

Construct a knowledge graph oriented to the crowdsensing system for social network users and filter to obtain user attributes related to the system, mainly including {"location," "interest type," "execution time," "willingness to participate"}. Set user's interest task types including {"air quality," "environmental monitoring," "sports," "movie," "politics"}. Calculate the matching degree of the subusers recruited by the participating users with the system tasks and their willingness to participate, analyze the user's influence, and obtain the user's social benefits. The greater the user's influence is, the greater the social benefits the user brings.

$K = <\text{concept, instance, relationship, attribute, } rule>$, where K is used to represent the knowledge group.

Concept $= \{\text{concept}_i, i = 1, \ldots, n\}$. This concept is a set of abstract ontology, such as mobile nodes, tasks, and geographic locations.

Instance $= \{\text{instance}_i, i = 1, \ldots, m\}$. This instance is a set of specific examples, such as user i and task t.

Attribute $= \{ < instance_i, \text{properties}_{ij}, \text{value}_j > \}$. These attributes are a set of instance attribute values, such as task {<content: taking a photo> | <description: photo format description> | <location: perception location> | <time: task timeliness>}.

Relationship $= \{ < \text{concept}_i, \text{relation}_{cc}, \text{concept}_j > | \langle \text{concept}_i, \text{relation}_{ci}, \text{instance}_j \rangle | \langle \text{instance}_i, \text{relation}_{ii}, \text{instance}_j \rangle \}$. Relationship shows the relationship between instances, such as < user, choice, task>.

Rule $= \{\text{rule} | \text{rule} = < \text{instance}_i, \text{new relation}_{ij}, \text{instance}_j > | < \text{concept}_i, \text{new relations}_{ij}, \text{instance}_j > | < \text{instance}_i, \text{properties}_{ij}, \text{new value}_j > \text{based on } K\}$. This rule is used to derive new attributes and new relationships. The rule in this article derives new attributes as $\langle \text{ure, inofluence, value} \rangle$.

Relationship inference uses existing relationship instances to infer new relationships between instances. As shown in Figure 3, assuming that the system has five tasks to be performed, Li Ming only recruited Zhang San to participate in the perception task by sharing the perception task. Their task tendencies were different. Zhang San chose the "air quality" task and Li Ming chose the "environmental monitoring" task. They were in the same location as the "park," and their respective willingness to participate was also different. According to the knowledge graph to match tasks, Zhang San's task matching degree was $Ma = 1$ and then Zhang San's potential contribution was $\mu = 0.6$. The influence (influence$_i$, value) of users N_i is derived based on the attributes and task selection rules among users. In this article, influence$_i$ is the influence of the user N_i, and a_i^t.

When calculating the influence of a user, it is not enough to only consider the user entity connected to the user. The attribute characteristics of the user should also be taken into account. The matching degree between recruited subusers and system tasks has a direct impact on superior users.

3.3. Optimize Social Welfare. Based on the crowdsensing task specification and the attribute inference of the knowledge graph for crowdsensing, the following questions are raised. Given a task $T = \{\tau_i, \ldots, \tau_m\}$ of a set of spatial units, a group of mobile users $N = \{n_1, n_2, \ldots\}$ on a social network have check-in data (including location) in the task space, and the user's task coverage is used to characterize the perceived quality $v_i = \text{Covered}(n_i)$. After the platform releases the task set T, sensing users who want to participate can increase their own social benefits by recruiting their friends to participate in the sensing task and further increase their probability of being selected and their reward after being selected. After the platform receives the user's bid, it

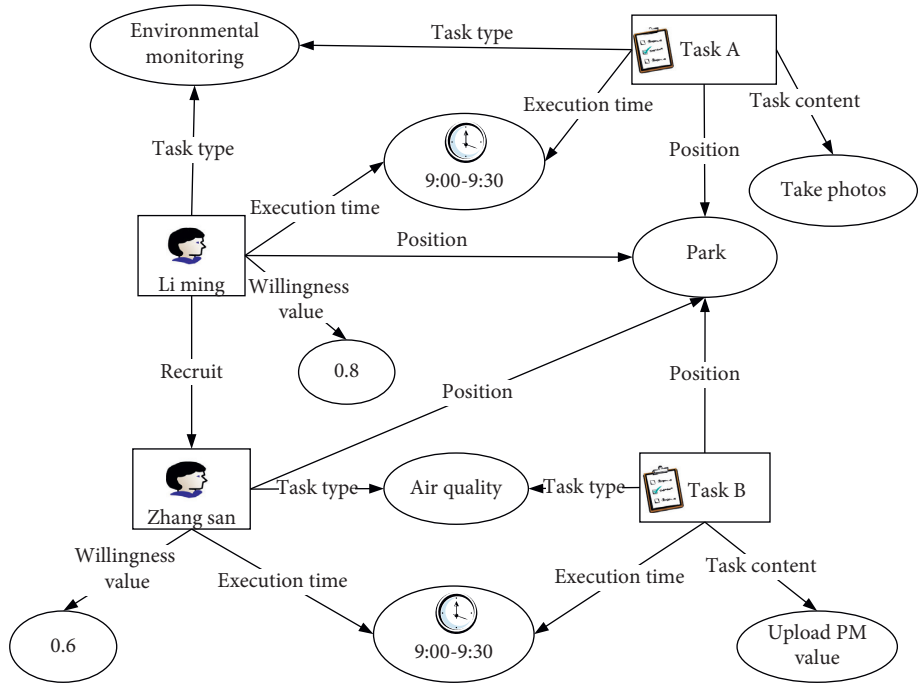


Figure 3: Example of user influence derivation.

analyzes the user attribute inference generated by the knowledge graph, the user's location, and the user's bid, selects the winner user, and, after the platform receives the user's sensing task data, pays the winner user the corresponding reward. Users who are not selected do not perform any tasks and are not paid. Based on the above theory, the relevant definitions are as follows.

Definition 1 (user utility). *The utility of all user $i \in N$ is defined as*

$$u_i = \begin{cases} p_i - c_i, & i \in W, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where W is a set of winning users, c_i is the true cost of user i, and p_i is the remuneration obtained by user i.

Definition 2 (platform utility). *Platform utility is defined as the total value that the platform can obtain minus the total cost of user sensing tasks. v_i is the narrow value that users bring to the platform through sensing tasks. Se_i is the broad value that users bring to the platform through social relationships. S_i is the total value that users bring to the platform, that is, the sum of narrow value and broad value:*

$$u_0 = \sum_{i \in W} (v_i - p_i + Se_i) = \sum_{i \in W} (S_i - p_i). \tag{2}$$

Definition 3 (social welfare). Social welfare is defined as the overall benefit of the crowdsensing ecosystem, that is, the sum of platform utility and user utility:

$$Sw = \sum_{i \in W} (S_i - b_i). \tag{3}$$

In a real auction, the user's bid b_i is equal to the user's real cost c_i.

The single run auction (SRA) problem can be defined as in run r, and the platform hopes to maximize the objective function within a certain constraint. The problem in this article belongs to the SRA problem. The goal is to find the final winner user set to maximize social welfare. It can be expressed as

$$\begin{aligned} w^* &= \arg\max_{W \in N} \sum_{i \in W} (S_i - b_i), \\ s.t. &\sum_{i \in W} p_i \le B, \end{aligned} \tag{4}$$

where N is the set of users who compete to participate in the sensing task and W is the set of winner users. The constraint is that the sum of the rewards of all winner users does not exceed the platform budget B.

In this process, we consider that users who aim to maximize their own benefits will have strategic and selfish behaviors. Assuming that users are rational but selfish, their natural goal is to maximize their respective utility. They may choose to submit forged bids, which may increase their effectiveness. On the contrary, we value the overall benefits of the entire crowdsensing ecosystem and pursue the highest social welfare. For this reason, it is important to solicit real bids from users. We aim to design a dominant strategy mechanism in which each worker has a dominant strategy defined in Definition 4 [23].

Definition 4 (dominant strategy). *If and only if, for any other strategy St_i' and any strategy profile of other workers (denoted as St_{-i}), strategy St_i is the dominant strategy of worker i and satisfies $u_i(St_i, St_{-i}) \ge (St_i', St_{-i})$.*

In the auction process of this article, each worker submits a bid (Γ_i, b_i) to the platform, which includes the task set Γ_i and bid price b_i that she is interested in. Because the worker is of strategic importance, she may announce that the bid deviates from the actual value. One of the goals of the incentive mechanism designed in this paper is to design the real mechanism defined in Definition 5.

Definition 5 (authenticity of bid price). *The auction is real if and only if it is the main strategy for each worker $i \in N$ to bid for its true value* (Γ_i, c_i).

In addition to authenticity, another design goal is to ensure that each worker obtains nonnegative utility from participation. If the perception task submitted by the user is duplicated with the perception data that has been submitted, in order to avoid submitting redundant perception data, the perception platform will not pay for this part of the work, which may result in the user's remuneration becoming lower than the user's bid. To ensure the sensing user's personal benefits, the user will not participate in this perception task. This attribute is formally defined as personal rationality in Definition 6.

Definition 6 (user rationality). *The mechanism satisfies the user rationality if and only if each worker's reward is greater than the user's bid price, that is, $i \in N$ satisfies $u_i \geq 0$.*

4. Incentive Method Based on the Diffusion of Social Users

In view of the user's social network check-in and influence inference based on the knowledge graph, this article proposes a user-oriented incentive method to make user selection, and the author can choose the user who is most beneficial to the platform. Here, the benefits that users bring to the platform through tasks are defined as the sum of the narrow value that users bring to the platform through sensing tasks and the generalized value that users bring to the platform through social relationships. The algorithm will continue to select and add new users until the budget reaches the limit.

4.1. Social User Influence Definition. When constructing a knowledge graph for social users to infer the user influence of a user under a specific task, it is not enough to only consider the user entity connected to the user. The user's attribute characteristics should also be taken into account. By extending the PageRank algorithm, a user influence analysis based on attribute characteristics is obtained. The similarity of the user's attribute characteristics shows that the user is willing to establish contact and interact with people with the same interests and hobbies. The higher the similarity of the user is, the greater the possibility of being influenced by the other party.

The similarity of attribute characteristics between user V_{i}^U and user V_{j}^U can be regarded as the difference between two users for a certain attribute characteristic. If the distribution of two users on a certain attribute characteristic is very similar, the similarity will be greater. The similarity sim_{ij} between user V_{i}^U and user V_{j}^U on the task attribute feature is defined as

$$sim_{ij} = \cos\left(\overrightarrow{\theta}_i, \overrightarrow{\theta}_j\right), \tag{5}$$

where $\overrightarrow{\theta}_i$ is the task attribute vector of user V_{i}^U and $\overrightarrow{\theta}_j$ is the task attribute vector of user V_{j}^U.

Given the recruited user set N_i of user entity V_{i}^U and the probability of user V_{i}^U sharing the sensing task to user, $V_{\mathrm{j}}^U \in N_i$ is expressed as TP_{ij}:

$$TP_{ij} = \frac{1}{|N_i|}, \tag{6}$$

where TP represents the relationship transition matrix, TP_{ij} is the normalized value, and $|N_i|$ represents the number of users recruited by user V_{i}^U.

According to the task attributes provided by the perception platform, the influence of user V_{i}^U on user V_{j}^U is related to the similarity of attribute characteristics. The more similar the topic distribution of two social users, the greater the transition probability between users. The probability that user entity V_{i}^U spreads the sensing task and successfully recruits to user entity V_{j}^U is SP_{ij}:

$$SP_{ij} = TP_{ij} * sim_{ij}, \tag{7}$$

where SP is the similarity transition matrix, which represents the probability of each user successfully recruiting related users.

According to the given user entity relationship and the task attribute characteristics of the user entity, the relationship transition matrix TP is obtained, and the similarity transition matrix SP between users is obtained. The user influence analysis based on task attribute characteristics is defined as

$$\begin{aligned} R &= [a_i], \quad i \in N, \\ R &= \alpha * SP * R + (1-\alpha)E, \end{aligned} \tag{8}$$

where E is the probability matrix of randomly jumping to any user, $\|E\| = 1$, and α are a parameter between 0 and 1 that controls the probability of teleportation, usually $\alpha = 0.85$.

4.2. Single-Task Budget Constraints' User Choice. This paper considers the incentive mechanism of crowdsensing when perceiving users facing a single task under an ideal situation where user information is known. Through the analysis of user influence in Section 4.1, the social benefits brought to the platform by users' recruiting workers through social relationships are characterized as

$$Se_i = \sum_{j \in N_i} a_i v_j, \quad i \in W. \tag{9}$$

According to the sensing information submitted by users and their social contributions, the social welfare created by user i is $Sw_i = S_i - b_i$.

Definition 7 (submodule function). *Given function* Ψ: $2^N \longrightarrow R$, *if, for any set* $\forall X \subseteq \Delta \subseteq N, \quad \forall \chi \in N \backslash \Delta$, *there is*

$$\Psi(X \cup \{\chi\}) - \Psi(X) \geq \Psi(\Delta \cup \{\chi\}) - \Psi(\Delta), \tag{10}$$

then function Ψ is a submodular function.

Therefore, the social welfare function in definition three is a nonnegative submodule function. When there is only a single task, we only need to select the user with the largest social welfare created under the task in turn and calculate the rewards of the winner users in real time to meet the budget limits. If the user is ultimately not selected to participate in the perception task, then $p_k = 0$, otherwise the reward is

$$p_k = b_k + (Sw^* - Sw^*_{-k}), \tag{11}$$

where Sw^* is the maximization of the objective function when user i is involved and Sw^*_{-k} is the maximization of the objective function when user i is not involved. They are defined as follows:

$$\begin{aligned} Sw^* &= \max_{W \in N} \sum_{i \in W} (S_i - b_i), \\ Sw^*_{-k} &= \max_{W \in N} \sum_{i \in W} (S_i - b_i), \end{aligned} \tag{12}$$

where N_{-k} represents the user set that does not contain user k. Therefore, when facing a single task, we select the winning users based on the social welfare created by the users and pay compensation based on the difference in their contribution to complete the positive incentive.

4.3. User Selection Algorithm under Multitask Budget Constraints. Although the incentive mechanism based on a single task is easy to implement, there are serious problems in efficiency. When faced with multiple tasks issued by the platform, the computational efficiency is too high. In addition, in the process of calculating user compensation, it is necessary to repeatedly calculate the social welfare created by users, which is very expensive. Therefore, the problem of low efficiency makes this incentive mechanism unsuitable for adoption on large social networks. Therefore, this section proposes a social awareness incentive mechanism oriented to multitask greed, with the goal of maximizing system social welfare.

First, some definitions are introduced to clarify in detail the social perception incentive mechanism of greed, including user selection rules and payment rules.

Definition 8 (user contribution). *User contribution is defined as the sum of the total value of all tasks that user i can complete:*

$$V_i = \sum_{\tau \in \Gamma_i} S_i^{\tau}, \tag{13}$$

where Γ_i is the set of tasks that the user can complete.

Definition 9 (winner set contribution). *Given a group of winner users, denoted as W, the winner set contribution is the sum of the contributions of all the winner users:*

$$M(W) = \sum_{i \in W} V_i. \tag{14}$$

Definition 10 (user edge contribution). *Given a group of winner user set W, the edge contribution of user* $k \notin W$ *is the gain brought to the winner set contribution:*

$$M_k(W) = M(W \cup k) - M(W). \tag{15}$$

The marginal contribution of users is a key factor in the greedy society sensing and incentive mechanism. In the face of multitasking and multiuser, we propose a social awareness incentive mechanism based on greed in order to reduce the computational complexity. The platform selects a group of users to complete the sensing task. The goal is to maximize the social welfare of the system, which can be optimized as the following form:

$$\begin{aligned} w^* &= \arg\max_{W \in N} \left(\sum_{i \in W} \sum_{\tau \in \Gamma} S_i^{\tau} - \sum_{i \in W} b_i \right), \\ &= \arg\max_{W \in N} \left(\sum_{i \in W} V_i - \sum_{i \in W} b_i \right), \\ &s.t. \sum_{i \in W} p_i \leq B. \end{aligned} \tag{16}$$

The selection of users in SARA is based on the user's net contribution margin, and the ranking of the winner users is as follows:

$$M_{\Omega(1)} - b_{\Omega(1)} \geq \ldots \geq M_{\Omega(|W|)} - b_{\Omega(|W|)}, \tag{17}$$

where $\Omega(1)$ is the user index ranked first among the winner users. In SARA, ω_i is the user's net contribution margin and B is the budget constraint of the platform.

In order to calculate the remuneration paid to user J, sort according to the net contribution margin of users in N^+_{-j}. In each repeated loop, find the user with the largest net contribution margin and update the remuneration of user j to maximize it by replacing. At least one user in N^+_{-j} has been selected, making user j the winner user, where N^+_{-j} represents the set of net contribution margin not less than zero and does not include user j. The users in H are sorted in the order of decreasing net contribution margin as follows:

$$\widetilde{M}_{\varphi(1)} - b_{\varphi(1)} \geq \ldots \widetilde{M}_{\varphi(n)} - b_{\varphi(n)} \geq \ldots, \tag{18}$$

where $\varphi(n)$ is the index of the user at position n in the sequence and $\widetilde{M}_{\varphi(n)} = M(H \cup \{\varphi(n)\}) - M(H)$.

4.4. Algorithm Validity Analysis. When facing a single task, we aim to design a mechanism that maximizes social welfare while ensuring authenticity. In this part, we prove that the proposed incentive method satisfies the authenticity of user bidding and user rationality. The authenticity of user bids means that the user's utility will not be improved by users' submitting bids that deviate from their true cost prices. Before proving the authenticity, first rewrite the user's utility formula:

$$\begin{aligned} u_k &= p_k - c_k, \\ &= b_k + \left(Sw^* - Sw^*_{-k}\right) - c_k. \end{aligned} \tag{19}$$

Lemma 1. *The incentive mechanism proposed is to satisfy the authenticity of user bidding.*

Proof. When user k is honest, there is $b_k = c_k$; when user k is dishonest, namely, $b_k \neq c_k$, there are the following two situations.

(i) Case 1 ($b_k > c_k$): in this case, we consider the following three cases.

- (a) Case 1.1 : when the user is honest, user k is the winner; when the user submits $b_k > c_k$, user k is still the winner.
  In this case, when the user is honest, the user utility is $u_k = Sw^* - Sw^*_{-k}$, and when the user is dishonest, the user utility is $\widetilde{u}_k = \widetilde{S}w^* - \widetilde{S}w^*_{-k} + b_k - c_k$. Since the user set of winners that does not include user k is constant regardless of whether user k is honest or not, so $\widetilde{S}w^*_{-k} = \widetilde{S}w^*_{-k}$, and $u_k - \widetilde{u}_k$ can be calculated as

$$\begin{aligned} u_k - \widetilde{u}_k &= Sw^* - \widetilde{S}w^* - b_k + c_k, \\ &= \left(\sum_{i\in W} S_i - c_i\right) - \left(\sum_{i\in W} S_i - c_i + c_k - b_k\right) \\ &\quad - b_k + c_k = 0, \end{aligned} \tag{20}$$

  where the utility gain of user k is 0, so the user's income cannot be improved in this case.
- (b) Case 1.2 : When the user is honest, user k is the winner; when the user submits $b_k > c_k$, user k is not the winner. In this case, the user benefit when the user is dishonest, $\widetilde{u}_k = 0$, and when the user is honest, the user benefit is

$$\begin{aligned} u_k &= p_k - c_k, \\ &= \left(Sw^* - Sw^*_{-k}\right) + b_k - c_k, \\ &= Sw^* - Sw^*_{-k}, \\ &\geq 0. \end{aligned} \tag{21}$$

  Therefore, $u_k \geq \widetilde{u}_k$.
- (c) Case 1.3 : when the user is honest, user k is not the winner; when the user submits $b_k > c_k$, user k is not the winner. In this case, k.

(ii) Case 2 ($b_k < c_k$): in this case, we consider the following three cases.

- (a) Case 2.1 : when the user is honest, user k is the winner; when the user submits $b_k < c_k$, user k is still the winner. The proof of this case is the same as Case 1.1.
- (b) Case 2.2 : when the user is honest, user k is not the winner; when the user submits $b_k < c_k$, user k is the winner. In this case,

$$\begin{aligned} \widetilde{u}_k &= \widetilde{S}w^* - \widetilde{S}w^*_{-k} + b_k - c_k, \\ &= \widetilde{S}w^* - Sw^* + b_k - c_k, \\ &= \sum_{i\in W} S_i - \sum_{i\in W, i\neq k} c_i - b_k + b_k - c_k, \\ &= \sum_{i\in W} S_i - \sum_{i\in W} c_i - Sw^* \leq 0, \end{aligned} \tag{22}$$

  where $\sum_{i\in W} S_i - \sum_{i\in W} c_i$ is the social welfare when the winner user set includes user k, which is obviously smaller than the optimal social welfare Sw^* when user k is honest. In this case, user k's social benefits will not increase.
- (c) Case 2.3 : when the user is honest, user k is not the winner; when the user submits $b_k < c_k$, user K is not the winner. In this case, $u_k = \widetilde{u}_k = 0$. □

Lemma 2. *This incentive mechanism satisfies user rationality.*

Proof. Personal rationality means that the user's income cannot be negative. For users who are not selected, their income is zero; for the winner users who participate in the perception task, their income is

$$\begin{aligned} u_k &= p_k - c_k, \\ &= Sw^* - Sw^*_{-k} \geq 0. \end{aligned} \tag{23}$$

In summary, the incentive mechanism based on social relationships meets the authenticity of bidding and user rationality.

A greedy social awareness incentive mechanism is proposed for multitasking. Here, it will be proved that it satisfies user rationality, bid authenticity, and computational efficiency. Before the proof, first introduce some symbols and properties for convenience. (i) $\Omega(\bullet)$ represents the function mapping of the user index when sorted in the descending order. (ii) $M_{n(n)} = M(W_{n-1} \cup \{\Omega(n)\}) - M(W_{n-1})$ represents the marginal contribution value of $\Omega(n)$. □

Lemma 3 (user rationality). *The proposed mechanism satisfies user rationality.*

Proof. For the n user in the formula, $M_{n(n)} - b_{\Omega(n)} \geq M_{\varphi(n)} - b_{\varphi(n)}$ is satisfied, so $M_{n(n)} - (M_{\varphi(n)} - b_{\varphi(n)}) \geq b_{\Omega(n)}$. At the same time, there is $p_{\Omega(n)} \geq M_{n(n)} - (M_{\varphi(n)} - b_{\varphi(n)})$ in the algorithm. So, the utility of each user is nonnegative, namely, $p_{\Omega(n)} \geq b_{\Omega(n)}$. Therefore, the greedy social awareness incentive mechanism is rational for users. □

Lemma 4 (authenticity of bids). *The proposed mechanism satisfies the authenticity.*

Proof. The auction mechanism is real, and if and only if the selection rules are monotonous, the reward for each winning user is a critical value. The monotonous selection rule is that when a user wins the task qualification with bi, he can still win the task qualification with a bid lower than bi. The compensation threshold means that when the user's bid is higher than this threshold, the user will not be selected.

Therefore, it is only necessary to prove that the user's selection rule is monotonous, and the reward of each winning user is a critical value, and then, it can be proved that the proposed incentive mechanism satisfies the authenticity.

Monotonic: *user selection is monotonous.* □

Proof. When a user bids with a lower bid, the user's net contribution margin will increase so that the user advances in the ranking and has a greater chance of being selected as the winner. When a user wins by bidding b_i, he can still win by bidding with a bid lower than b_i. Therefore, the user's selection rule is monotonous.

Critical value of payment: *the remuneration paid to each user is the critical value.* □

Proof. According to the payment rules, $p_{\Omega(n)} = \max_{\omega_m \geq 0, B - \sum_{j-1}^{n-1} p_j \geq 0} \{M_{n(m)} - (\tilde{M}_{\varphi(m)} - b_{\varphi(m)})\}$ can be obtained. When $b_{\Omega(n)} > p_{\Omega(n)}$, there is $b_{\Omega(n)} > M_{n(m)} - (\tilde{M}_{\varphi_m} - b_{\varphi_m})$ for all $\omega_m \geq 0$, namely, $M_{n(m)} - b_{\Omega(n)} < \tilde{M}_{\varphi(m)} - b_{\varphi(m)}$. $\Omega(n)$ will be after m in the descending order of net contribution margin, that is, $\omega_{\Omega(n)} < 0$; then, $\Omega(n)$ will not be selected as the winner user. Therefore, $p_{\Omega(n)}$ is the critical reward of user $\Omega(n)$.

By proving the monotonicity of the user selection rules and the critical value of payment, it can be concluded that the proposed incentive mechanism satisfies the authenticity. □

Lemma 5 (computational efficiency). *The proposed multi-task greedy incentive mechanism is computationally efficient.*

Proof. The computational efficiency means that the user selection and payment rules of the proposed mechanism can be solved in polynomial time. Assuming that there are N sensing users, in user selection, the time it takes to find the user with the largest net contribution margin does not exceed $O(N^+)$, and the time it takes to calculate each contribution margin does not exceed $O(N^2)$, in each while loop (line 2–12 in Algorithm 1). When calculating user compensation, in the repeat loop (Lines 5–9 in Algorithm 1), the time spent calculating each contribution margin does not exceed $O(N^2)$, and the time it takes to find the user with the largest net contribution margin does not exceed $O(N^+)$. □

5. Experimental Verification and Analysis

In this section, we introduce the baseline method used in the simulation as well as the simulation settings and results. First, compare SARA with the baseline to prove its competitiveness. Then, verify SARA's user rationality, budget feasibility, and authenticity of user bids.

5.1. Baseline Method and Simulation Settings. The goal of the method proposed in this paper is to maximize social welfare under budget constraints, so the baseline method should also meet the budget constraints. The first baseline method is a modified version of the QoI-SRC auction [14], which integrates budget constraints into the QoI-SRC winner determination problem. The QoI-SRC auction has proved to be close to the best social welfare, and the improved QoI-SRC auction is called the SRC auction. Another baseline method is OPT [24], which represents the estimated upper limit of the best solution to the SRA problem when the platform fully understands the user's real bid. Since calculating the best solution to the SRA problem is very time-consuming and cannot be calculated in a reasonable time as the problem size grows, we use the best upper limit instead of the best itself as the benchmark.

Aiming at the impact of the number of users, budget, and number of tasks on social welfare, the experimental parameters are set in Table 1 to simulate the SRA problem, and experimental observations are made. In setting I, the fixed number of tasks is 500, and the budget is set to 600 and 800. The number of users is used as an independent variable from 100 to 300. In setting II, the number of fixed tasks is set to 100 and 200. In this case, the budget is used as an independent variable from 100 to 500. In setting III, the fixed budget constraint is 200, and the user sets 100 and 200. The number of tasks is used as an independent variable from 100 to 500. Among them, the sensing data quality q_i provided by the user, the sensing cost c_i, the task quality requirement Q_j, and the task set $|\Gamma_i|$ selected by the user will be generated uniformly and randomly selected from the range given in Table 1. Among them, the maximum task set Γ_i that the user can perform is $|\Gamma_i|$ tasks randomly selected from T.

Table 2 shows the experimental settings for user rationality, budget feasibility, and bid authenticity verification. Set the number of users, budget and tasks given in IV, and observe whether the user's income and expenditure under this setting meets the user's rationality. In setting V, given the number of users and the number of tasks, use the budget as an independent variable to observe whether the relationship between the true total expenditure and the budget meets the budget feasibility under this setting. In setting VI, given the number of users, budgets, and tasks, a winning user and a failed user are randomly selected on this basis, and the profitability of submitting different bids is observed, and the graph shows whether the bid authenticity is satisfied. The parameter settings in Tables 1 and 2 are not deliberately selected, so the experimental results will not be lost. And, in Tables 1 and 2, we only consider settings with continuous quality requirements.

5.2. Simulation Results. The social welfare comparison of SARA, SRC, and OPT is shown in the figure. It can be observed that social welfare increases with the increase in the number of users when the budget is fixed, until the budget

(1) $W \leftarrow \varnothing$, $p_h \leftarrow 0$;
(2) **while** $\omega_i \geq 0 \&\& B \geq 0$ **do**
(3) $\quad j \leftarrow \arg\max_{h \in N^+ \setminus W} (M_h(W) - b_h)$;
(4) $\quad \mathrm{H} \leftarrow \varnothing$;
(5) repeat
(6) $\quad k \leftarrow \arg\max_{h \in N_{-j}^+ \setminus \mathrm{H}} (M_h(W) - b_h)$;
(7) $p_j \leftarrow \max\{p_j, \min\{M_j(\mathrm{H}) - (M_k(\mathrm{H}) - b_k), M_j(\mathrm{H})\}\}$;
(8) $\quad \mathrm{H} \leftarrow \mathrm{H} \cup \{k\}$;
(9) **until** $N_{-j}^+ \setminus \mathrm{H} = \varnothing$;
(10) $\quad W \leftarrow W \cup \{j\}$;
(11) $\quad B \leftarrow B - p_j$;
(12) **end while**
(13) **return** $W, p_j, \forall j \in W$;

Algorithm 1: SARA greedy social awareness incentive.

Table 1: Parameter settings.

	q_i	c_i	Q_j	$\lvert\Gamma_i\rvert$	N	B	T
I	[2, 4]	[4, 8]	[10, 13]	[20, 30]	[100, 300]	600, 800	500
II	[2, 4]	[4, 8]	[10, 13]	[20, 30]	100, 200	[100, 500]	500
III	[2, 4]	[4, 8]	[10, 13]	[20, 30]	100, 200	200	[100, 500]

Table 2: Parameter settings.

	q_i	c_i	Q_j	$\lvert\Gamma_i\rvert$	N	B	T
IV	[2, 4]	[4, 8]	[10, 13]	[20, 30]	500	1000	500
V	[2, 4]	[4, 8]	[10, 13]	[20, 30]	300	[0, 1000]	500
VI	[2,4]	[4, 8]	[10, 13]	[20, 30]	100	500	500

limit reaches the saturation level from Figure 4. Among them, OPT is the best solution to the SRA problem. From Figure 4, it can be concluded that SARA is close to the best social welfare and far higher than the social welfare of the baseline SRC under the same number of users. SRC performed the worst among the three, mainly because it was over-reliant on the budget and was limited by the budget. Therefore, the social welfare generated by SRC in Figure 4 does not increase with the increase in the number of users. However, it can be clearly seen in Figure 5 that it is linearly increasing. However, the number of users does not affect it, mainly, because of its excessive dependence on budget costs.

In Figure 5, it can be observed that social welfare increases with the increase of the budget when the number of users is fixed. When the number of users is 100 and the budget is 450, the social welfare of SARA and OPT tends to be stable. When the budget reaches 450 and the number of users is 200, the social welfare of SARA and OPT is still steadily increasing compared with the number of users of 100. SRC increases as the budget increases, but it is far lower than the social welfare of SARA and OPT. This is because SRC chooses to perceive users based on the initial marginal social welfare effectiveness to select new winners, rather than updating the effective marginal contribution in each iteration such as SARA.

Figure 6 shows that the social welfare of SARA, SRC, and OPT increases with the number of tasks. For SARA and OPT, the budget is sufficient under this setting, but SRC is growing slowly due to budget constraints. Among them, SARA produces social welfare after the number of tasks reaches 200 and gradually approaches the best social welfare of OPT. The main reason is that the set of tasks selected by users in SARA is large. In the case of a small number of tasks, users cannot meet their balance of expenditures. It can be seen that the current SARA is suitable for larger-scale problems.

User rationality, as shown in Definition 6, requires that the labor remuneration obtained by the user through the sensing task is greater than the user cost submitted by the user to ensure that the user's benefit is positive. The experiment is set to IV in Table 2, fixed number of users, budget, and number of tasks. For each winning user, the total cost and the reward he received are plotted in Figure 7. In Figure 7, the abscissa is the serial number of the user who won the bid, the plus sign represents the reward that the user receives after completing the sensing task, and the circle

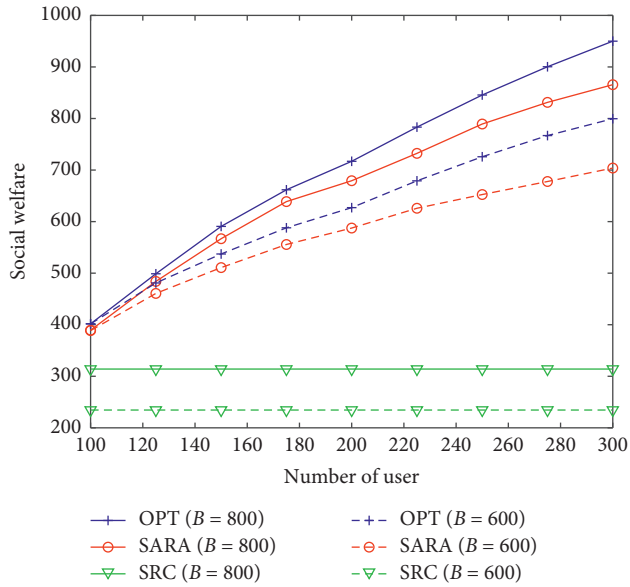


Figure 4: The impact of the number of users on social welfare.

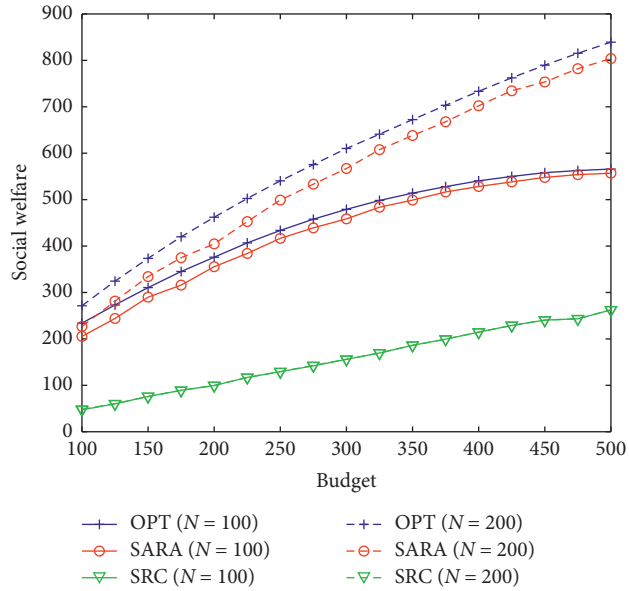


Figure 5: The impact of the budget on social welfare.

represents the user's bid price. Obviously, the user's reward is always greater than the corresponding total cost. It proves the user rationality of SARA.

Budget feasibility requires that the total expenditure paid by the platform to sensing users cannot exceed the platform's budget, and the task should be completed within the budget. The experiment is set to V in Table 2. The number of users and the number of tasks are fixed, and the budget is used as an independent variable to vary from 0 to 1000 with a step length of 100. We observe the total expenditure of this mechanism under different budgets. For each fixed budget, we calculate the actual total payment of the requester and display the result in Figure 8. From Figure 8, we can understand that the total payment is close to the budget when the budget is small. Due to the limitation of the number of workers, the total payment will reach saturation level when the budget is large. Obviously, the total payment never exceeded the budget, which proved the budget feasibility of SARA.

The authenticity of user bids, as shown in Definition 5, requires that regardless of whether the winner submits a lower or higher bid than the actual bid, it will not increase its own profit, and the loser who submits lower than the real bid will not become a winning user. This means that

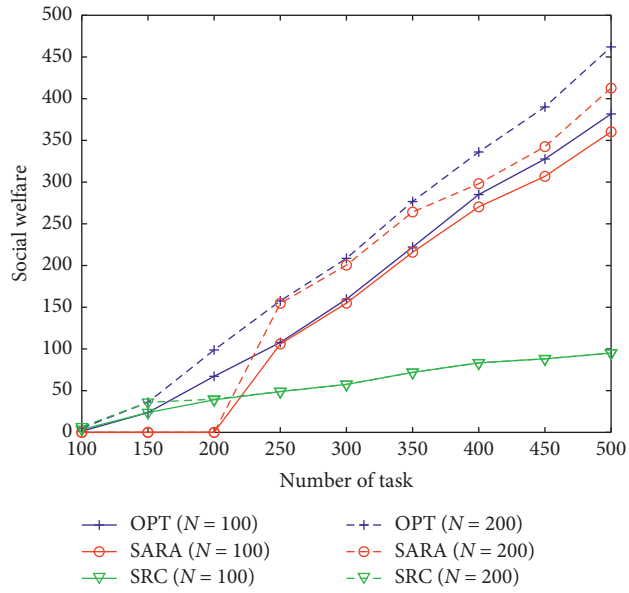


Figure 6: The impact of the number of tasks on social welfare.

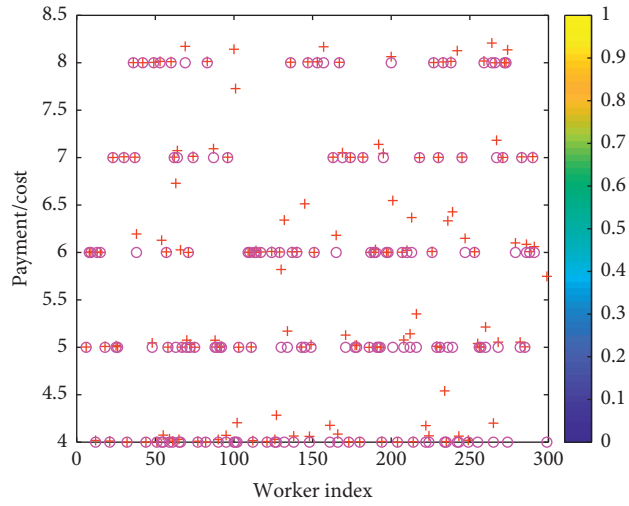


Figure 7: SARA user rationality test.

the user's false bid will not change the user's own income. Given the parameter settings of VII in Table 2, fixed number of users, budget, and number of tasks, we randomly select winner with a true bid of 7 and loser with a true bid of 7 from the set of all workers and use the actual bids of the costs of winner and loser to understand how their utility changes. The cost authenticity test of winner and loser is shown in Figure 9. In these several situations, we can observe that when a worker bids for its true value, its utility is maximized. Regardless of whether the user is a

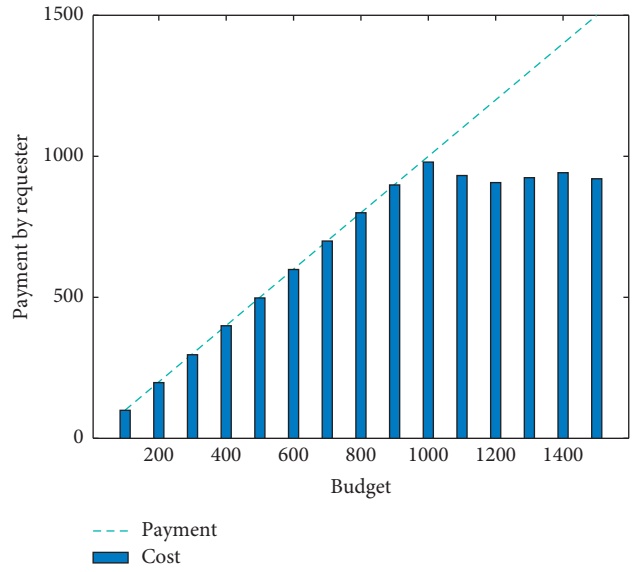


Figure 8: Budget feasibility verification.

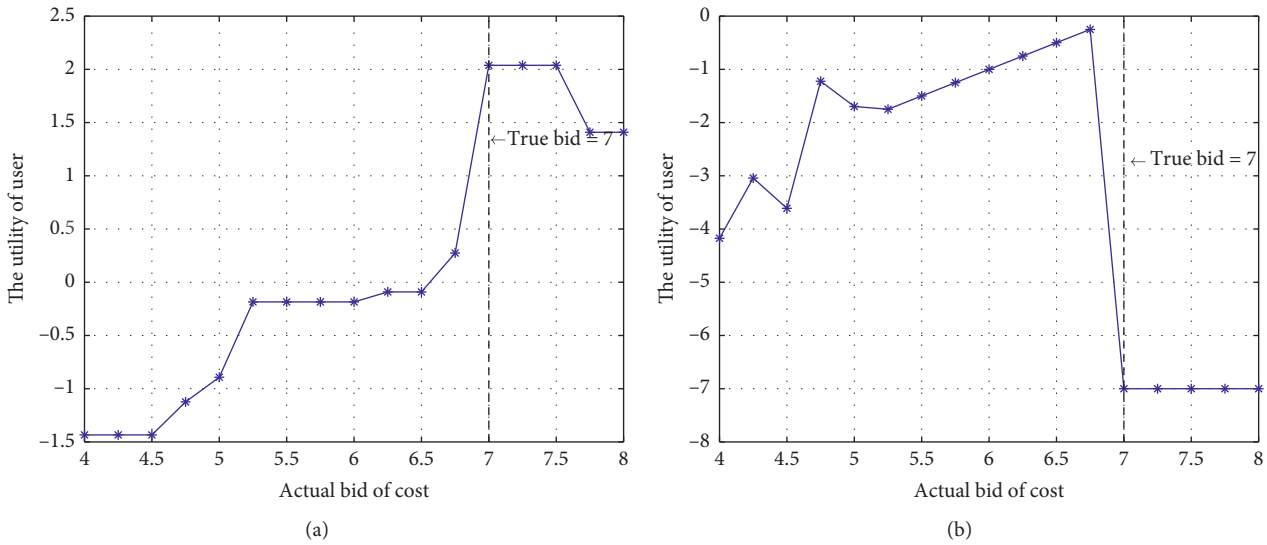


Figure 9: Authenticity verification of SARA. (a) Cost authenticity of winner. (b) Cost authenticity of loser.

winner or a loser, the user cannot increase its utility through dishonest bids, which proves the authenticity of SARA's bidding.

6. Conclusion

In this paper, an incentive method called SARA is proposed, which is a social awareness incentive mechanism of MCS system based on reverse auction. Different from the traditional MCS platform recruiting sensing users, social networks are used as the recruitment platform. The sensing task is diffused in the social network through the initial user, and the knowledge graph is used to comprehensively analyze the user's potential value and influence to obtain the user's social benefits. The user's social benefits and actual sensing data are taken as the overall contribution of the sensing users, and the sensing users with the greatest social welfare are selected as the winner users. Comparing the method proposed in this article with the existing literature and the optimal task-solving method, experiments show that, under the influence of various factors, the social welfare of SASR is much higher than that of SRC and close to the social welfare of OPT. At the same time, it proved that SASR satisfies the requirements of dominant strategy, user's personal rationality, bid authenticity, and computational efficiency.

In the future work, the application of crowdsensing under the knowledge map will be further explored. Use the complex semantic associations between entities provided by

the knowledge graph to recommend matching perception tasks for perception users. This has the effect of improving the performance indicators of crowdsensing. In addition, we only considered the uniform distribution of the cost of the sensing task and the user bid in the comparison with other solutions in the simulation process. Later, a comparative analysis under normal distribution and exponential distribution will be introduced.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This present research work was supported by the National Natural Science Foundation of China (61403109 and 61202458), Specialized Research Fund for the Doctoral Program of Higher Education of China (20112303120007), Heilongjiang Natural Science Foundation (LH2020F034), China Postdoctoral Science Foundation (2019M651263), and Scientific Research Fund of Heilongjiang Provincial Education Department (12541169).

References

[1] Y. Cheng, X. Li, Z. Li et al., "AirCloud: a cloud-based air-quality monitoring system for everyone," in *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pp. 251–265, Association for Computing Machinery, New York, NY, USA, November 2014.

[2] S. Hu, L. Su, H. Liu, H. Wang, and T. F. Abdelzaher, "SmartRoad," *Acm Transactions on Sensor Networks*, vol. 11, no. 4, pp. 1–27, 2015.

[3] R. K. Ganti, N. Pham, H. Ahmadi et al., "GreenGPS: A participatory sensing fuel-efficient maps application," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, pp. 151–164, Association for Computing Machinery, San Francisco, CA, USA, June 2010.

[4] A. Yadav, B. Wilder, E. Rice et al., "Influence maximization in the field: the arduous journey from emerging to deployed application," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 150–158, International Foundation for Autonomous Agents and Multiagent Systems, São Paulo, Brazil, May 2017.

[5] Y. Wang, G. Cong, G. Song et al., "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1039–1048, Association for Computing Machinery, Washington, DC, USA, July 2010.

[6] G. Song, X. Zhou, Y. Wang, and K. Xie, "Influence maximization on large-scale mobile social network: a divide-and-conquer method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1379–1392, 2015.

[7] Z. Han, Y. Chen, W. Liu et al., "Research on node influence analysis in social networks," *Journal of Software*, vol. 28, no. 1, pp. 84–104, 2017.

[8] Y. Quan, Y. Jia, L. Zhang et al., "Performance analysis and testing of personal influence algorithm in online social networks," *Journal on Communications*, vol. 39, no. 10, pp. 1–10, 2018.

[9] Y. Wu, J. Zeng, H. Peng et al., "Survey on incentive mechanisms for crowd sensing," *Journal of Software*, vol. 27, no. 8, pp. 2025–2047, 2016.

[10] Z. Zheng, F. Wu, X. Gao, H. Zhu, S. Tang, and G. Chen, "A budget feasible incentive mechanism for weighted coverage maximization in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 16, no. 9, pp. 2392–2407, 2017.

[11] R. Zhou, Z. Li, and C. Wu, "A truthful online mechanism for location-aware tasks in mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 8, pp. 1737–1749, 2018.

[12] D. Peng, F. Wu, and G. Chen, "Data quality guided incentive mechanism design for crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 2, pp. 307–319, 2018.

[13] C. Jiang and L. Gao, "Mechanism design for crowd sensing with data reuse based on two-sided auction," *Chinese Journal on Internet of Things*, vol. 3, no. 3, pp. 27–33, 2019.

[14] H. Jin, L. Su, D. Chen et al., "Thanos: incentive mechanism with quality awareness for mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1951–1964, 2018.

[15] H. Jin, L. Su, D. Chen et al., "Quality of information aware incentive mechanisms for mobile crowd sensing systems," in *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 167–176, Association for Computing Machinery, Hangzhou, China, June 2015.

[16] J. Xu, Z. Rao, L. Xu, D. Yang, and T. Li, "Incentive mechanism for multiple cooperative tasks with compatible users in mobile crowd sensing via online communities," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1618–1633, 2020.

[17] L. Duan, T. Kubo, K. Sugiyama, J. Huang, T. Hasegawa, and J. Walrand, "Motivating smartphone collaboration in data acquisition and distributed computing," *IEEE Transactions on Mobile Computing*, vol. 13, no. 10, pp. 2320–2333, 2014.

[18] J. Nie, J. Luo, Z. Xiong, D. Niyato, and P. Wang, "A Stackelberg game approach toward socially-aware incentive mechanisms for mobile crowdsensing," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 724–738, 2019.

[19] Y. Zhao, W. Song, and Z. Han, "Social-aware data dissemination via device-to-device communications: fusing social and mobile networks with incentive constraints," *IEEE Transactions on Services Computing*, vol. 12, no. 3, pp. 489–502, 2019.

[20] J. Sun, F. Hou, S. Ma et al., "Social-aware incentive mechanism for participatory sensing," in *Proceedings of the IEEE Global Communications Conference*, IEEE, Washington, DC, USA, December 2016.

[21] H. Long, S. Zhang, Y. Zhang et al., "Task distribution algorithm based on community in mobile crowd sensing," *Journal on Communications*, vol. 40, no. 10, pp. 43–54, 2019.

[22] J. Wang, F. Wang, Y. Wang, D. Zhang, L. Wang, and Z. Qiu, "Social-network-assisted worker recruitment in mobile crowd sensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 7, pp. 1661–1673, 2019.

[23] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, MIT Press, Cambridge, MA, USA, electronic edition, 1994.

[24] H. Wang, S. Guo, J. Cao, and M. Guo, "MeLoDy: a long-term dynamic quality-aware incentive mechanism for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 4, pp. 901–914, 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6616239, 13 pages
https://doi.org/10.1155/2021/6616239

Research Article

Robust Frame Duplication Detection for Degraded Videos

Qi Han,[1] **Hao Chen**,[1] **Liyang Yu**,[2] **and Qiong Li**[1]

[1]*School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China*
[2]*School of Software Engineering, Harbin University and Science and Technology, Harbin 150080, China*

Correspondence should be addressed to Qi Han; qi.han@hit.edu.cn

Received 31 December 2020; Revised 16 March 2021; Accepted 8 April 2021; Published 21 April 2021

Academic Editor: Liguo Zhang

To detect frame duplication in degraded videos, we proposed a coarse-to-fine approach based on locality-sensitive hashing and image registration. The proposed method consists of a coarse matching stage and a duplication verification step. In the coarse matching stage, visually similar frame sequences are preclustered by locality-sensitive hashing and considered as potential duplication candidates. These candidates are further checked by a duplication verification step. Being different from the existing methods, our duplication verification does not rely on a fixed distance (or correlation) threshold to judge whether two frames are identical. We resorted to image registration, which is intrinsically a global optimal matching process, to determine whether two frames coincide with each other. We integrated the stability information into the registration objective function to make the registration process more robust for degraded videos. To test the performance of the proposed method, we created a dataset, which consists of 3 subsets of different kinds of degradation and 117 forged videos in total. The experimental results show that our method outperforms state-of-the-art methods for most cases in our dataset and exhibits outstanding robustness under different conditions. Thanks to the coarse-to-fine strategy, the running time of the proposed method is also quite competitive.

1. Introduction

With various nonlinear editing tools such as Adobe Premiere, Microsoft Movie Maker, and Sony Vegas, it is now much easier for people to tamper the content of a video. Many different kinds of detection methods have been proposed [1–3]. Among different approaches to video forgery, frame duplication, which simply copies a sequence of frames to another position in the timeline, may be one of the most convenient yet effective means to hide or counterfeit events. The forged part of the video can be easily made visually natural and therefore difficult for manual detection. Fortunately, since the source and target frames simultaneously exist in the video, frame duplication forgery can be exposed by detecting abnormal identical frame sequences. On this basis, several methods have been proposed [4–8]. These methods share a common methodology to judge whether a frame is the copy of another one. These methods share a common methodology to judge whether a frame is the copy of another one. They extract features from the frames and set the distance threshold between the features. Such methodology makes it difficult for these methods to perform robustly when applied in realistic frame duplication detection (FDD), where degradation is quite common. In degraded videos, the local structure of the frames can be altered slightly by sorts of factors, and then the value of the extracted features correspondingly changes; therefore, a fixed distance threshold cannot always work well. For instance, an experienced attacker may add a little perturbation (e.g., additive noise) to either the source or the target frames; then, in this noisy scenario, it is quite probable that the threshold tuned for ordinary cases will miss some true matching frame pairs. In fact, even the lossy encoding process will result in substantial difference between the source and target frames; please see Figure 1 for an example. Since a tampered video will be subject to at least double compressions, this example indicates that degradation in video forgeries is almost unavoidable. This makes it quite complicated to stably detect frame duplication in realistic scenarios.

In this paper, we are attempting quite a different methodology for FDD. In the proposed method, we no longer rely on a fixed threshold to decide whether a video subsequence is the duplication of another. The key idea is

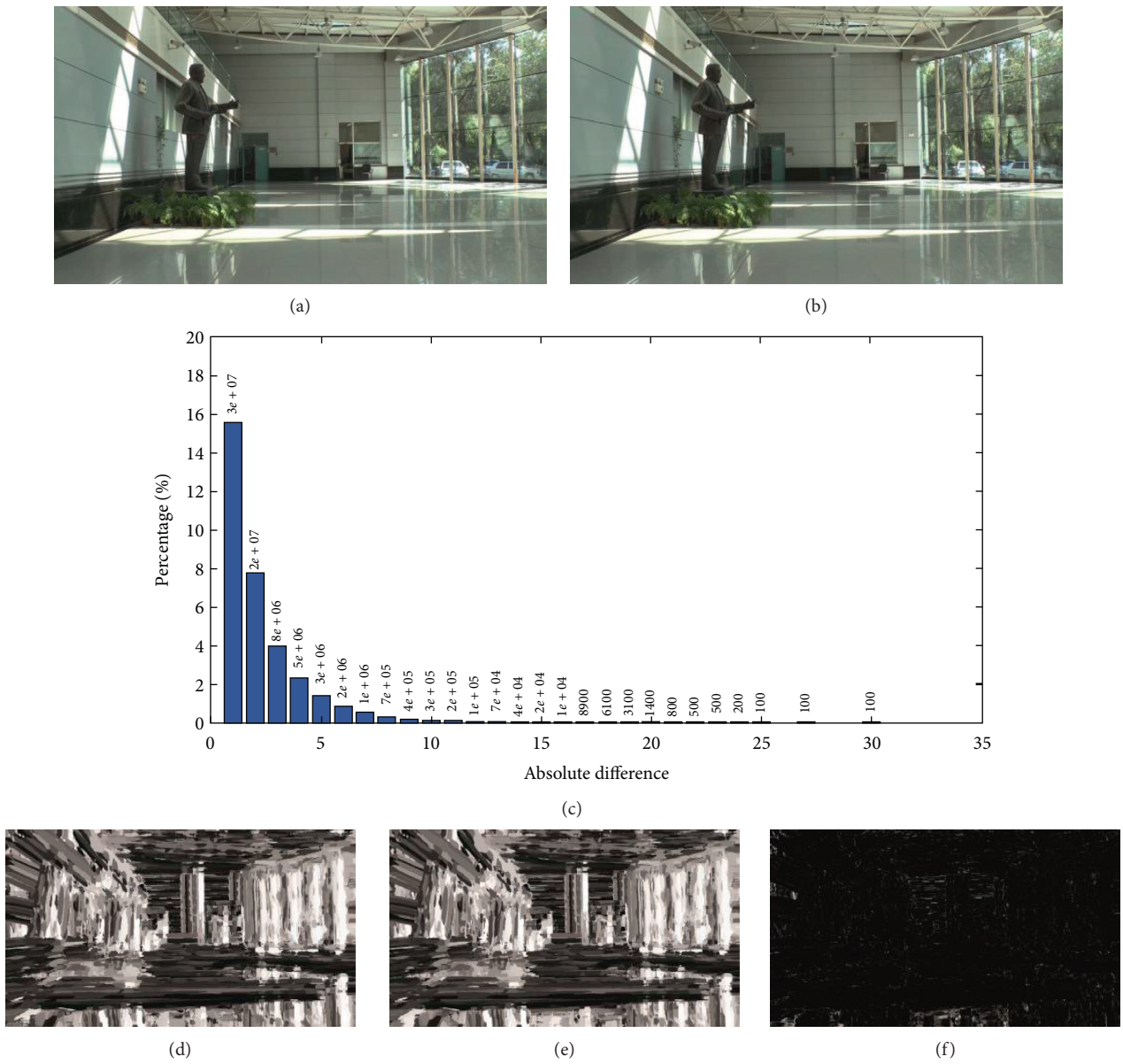


Figure 1: An example demonstrating the substantial difference between a frame and its copy. Such difference is caused only by the lossy encoding process itself. The video is H.264 encoded. (a, b) A pair of source and target frames. Although they are visually identical, nonnegligible error between them has been introduced during the encoding process. (c) The histogram of pixel-wise absolute difference. We can see that the error can even be as large as 30, and the number of occurrences of the errors which are larger than or equal to 10 is over 700,000 (the resolution of the video is 1920×1080). To further demonstrate the impact of such difference, we build a 500-word vocabulary by k-means clustering 7500 dense SIFT features extracted from 72 randomly selected images, and the vocabulary is lexicographically sorted so that neighbouring visual words in the vocabulary are also nearer in the feature space. We then extract dense SIFT features for each pixel in (a) and (b) and map the features to the indices of the visual words so that we get two visual word index maps (d, e) and the absolute difference map between them (f). The considerable amount of bright spots in (f) indicates that the lossy compression substantially changed the local structure and hence local feature.

that for any two frames f_a and f_b, if they contain the same objects, meanwhile the corresponding objects' shape and position are identical, f_a and f_b can be considered as the copy of each other. We resort to image registration to check whether the three aspects (objects, objects' shape, and objects' position) of two frames coincide with their counterparts. Specifically, the problem is solved by pixel-level global optimal matching. When a given frame is aligned to its copy, there should not be any distortion in the resulted offset field, i.e., each pixel in the source frame matches to the same location in the target frame. Our method is robust to certain magnitude of video degradation, but the global matching procedure is not so fast as some feature extraction and comparison-based methods such as [4]. For acceleration, our pipeline involves a coarse matching step, which can significantly improve the computational speed.

It should be noted that, like the other methods in this field [4–8], our method also does not take the static scenes into consideration. Therefore, our method can be used to expose counterfeit events. For instance, an attacker can copy a sequence of frames from the historical record to counterfeit the event that a man passing through a scene; then, the duplicated frames can be used as the evidence of being present or absent and therefore plant suspicion on that man or absolve him from guilty.

The contributions of this paper are listed as follows.

First, we propose a coarse-to-fine FDD scheme, whose first key step is preclustering perceptually similar sequences of frames by locality-sensitive hashing (LSH). Through this coarse matching step, the computation load for finer duplication verification can be reduced by several orders of magnitude.

Second, we use global optimal matching for finer duplication verification, with the benefit regarding computational cost obtained in the coarse matching step. We integrate the stability information of different regions into the matching objective, resulting in a robust yet sensitive matcher for the noisy environment.

The rest of this paper is organized as follows: in section "Related Work," we briefly introduce related work, and then in section "Proposed Method," the proposed method is detailed. The experimental results are presented in section "Experimental Results." The conclusion and future directions are finally drawn in section "Conclusion and Future work."

2. Related Work

To the best of our knowledge, there are only a few works on FDD. In [8], Wang and Farid proposed the first FDD method. The video is divided into overlapping short subsequences; then, for each subsequence, temporal correlation between each pair of frames and the block-wise spatial correlation within each frame are calculated and used as features for subsequence comparison. The subsequences are compared with each other in a coarse-to-fine manner. High similarity in temporal correlation coefficients will trigger the comparison between spatial correlation coefficients [9]. Given the number of frames, n, the time complexity of the subsequence comparison process is $O(n \cdot (n+1)/2)$. Methods using the same framework include [4, 6], which, respectively, use structural similarity (SSIM) [6] and histogram correlation to measure the similarity between subsequences.

Being different from [8], in [5, 7], the authors lexicographically sorted the features (Tamura texture and local binary pattern are used as features in [5, 7], respectively) corresponding to each frame; then, neighbouring features which are close enough to each other in the feature space correspond to duplicated frames. In this manner, the time cost spent on identifying matching frames is theoretically reduced to $O(k \cdot n \cdot \log n)$, where k is the feature dimension. It should be noted that, for lexicographical sorting, the features corresponding to the frames of the entire video have to be simultaneously stored in the memory. From an implementation perspective, in memory-constrained environments, with the increase of k and n, the storage needed by lexicographical sorting can sometimes exceed the memory capacity. In such cases, the features have to be stored on the disk instead; then, the sorting procedure will involve frequent disk access which is rather slow.

Taking the characteristic of the encoding process into consideration, Subramanyam and Emmanuel [10] extracted the histogram of oriented gradients (HOG) features from block pairs colocated in neighbouring I, B and P, B frame pairs; then, high correlation between the HOG features discloses duplicated blocks and hence frames. However, this method can only be used to detect the case that the source and target frames are placed next to each other, which is quite uncommon.

The methods discussed above unexceptionally detect the duplication behaviour by a fixed global threshold, which makes them less robust. Although some features can be, to some extent, robust to degradation, it is obvious that a threshold calibrated for a certain condition may not be suitable for others. This problem becomes particularly noticeable when processing degraded videos since there are too many unstable factors caused by, e.g., compression artifact or manually added noise.

One subject closely associated with FDD is near-duplicate identification (NDI) [11–14], whose major concern is the copyright issues. Unlike FDD, in NDI, the video clip in query is known, while the goal of FDD is to find all duplicated frame pairs within the video sequence where each frame can be possibly forged; therefore, FDD is more challenging in terms of time complexity, which increases quadratically with the number of frames [11]. Another major difference between FDD and NDI is that, in NDI, the potential attacks can be much stronger than in FDD, and the pirated videos can be geometrically transformed (e.g., picture in picture or recaptured videos) or inserted into logos or subtitles. In this sense, methods for NDI should be more robust while less discriminative than FDD.

3. Proposed Method

3.1. The Pipeline. We detect frame duplication in a coarse-to-fine manner. The pipeline consists of two key steps: coarse matching and duplication verification, as shown in Figure 2. The main concern of the coarse matching stage is to significantly reduce the burden of computation for the second stage. Given an input video V, following [8], we firstly divide V into $T - L + 1$ overlapping subsequences, where T is the total number of frames and L is the subsequence length. For each subsequence $t\,(1 \le t \le T - L + 1)$, using LSH, we identify from t's successive subsequences the ones that are visually similar to t. In this way, we cluster subsequence t and its duplication candidates into the same group. Then, in the duplication verification stage, we perform image registration between each pair of corresponding frames, respectively, in t and the duplication candidates. Through image registration, we obtain a series of offset fields, and the zero-offset fields verify duplications.

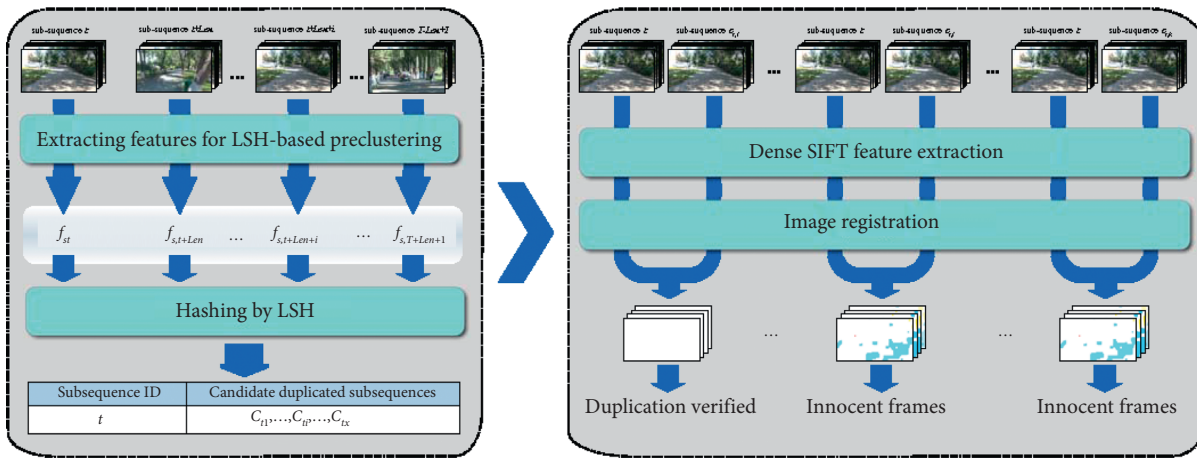


FIGURE 2: The pipeline of the proposed method. The pipeline consists of two key steps: coarse matching by LSH and duplication verification. This diagram depicts the process of identifying the duplication of a given subsequence t. Please see the text for details.

3.2. Coarse Matching by LSH. The left part of Figure 2 depicts the process of coarse matching for a given subsequence t. For subsequence t, we would like to find out, from subsequence $t+L$ to $T-L+1$, the duplication candidates $c_{t,1}, c_{t,2}, \ldots, c_{t,k}$ (where $k \le T-t-2\cdot(L-1)$), which are perceptually similar to t, such that the duplication verification procedure, which is more accurate but slower, can compare t only with $c_{t,i}\,(1 \le i \le k)$ instead of with all the succeeding subsequences of t. To this end, what we need is a feature which is sensitive to content (i.e., objects, objects' shape, and objects' position) change while robust against image degradation. Although many image hashing schemes can be used for this purpose (e.g., the wavelet-based [15] and SVD-based [16]), we find that a block-wise GIST [17] feature meets our requirement best in terms of the tradeoff between robustness, discriminative power, and computational time. For each frame, we extract the block-wise GIST descriptor; then, the descriptors extracted from the L frames in the subsequences t and $t+L+i\,(0 \le i \le T-L+1-t)$ are, respectively, concatenated to form one-dimensional features $f_{s,t}, f_{s,t+L+i}$ for the corresponding subsequences.

We exploit LSH to determine whether a feature $f_{s,t+L+i}$ is sufficiently close to $f_{s,t}$. Given an error probability $P_e > 0$ and a distance threshold R, when $\|f_{s,t} - f_{s,t+L+i}\| \le R$, LSH guarantees that $P_c \ge 1 - P_e$, where P_c is the collision probability for the hash value of $f_{s,t}$ and $f_{s,t+L+i}$. In this paper, we use the p-stable distribution-based LSH [18]:

$$h = \left\lfloor \frac{\mathbf{a}\cdot\mathbf{f} + b}{\omega} \right\rfloor, \tag{1}$$

where $\mathbf{f}$ is the feature vector being hashed, $\mathbf{a}$ is a real-valued vector whose elements are independently drawn from a standard normal distribution $N(0,1)$, which has been proven to be 2-stable [18] (since we use the l_2-norm to measure the difference between features), and b is a real scalar uniformly drawn from $[0, \omega]$, where ω is a real scalar.

To produce more reliable results, we construct H hash tables, and subsequence $t+L+i$ is considered as a duplication candidate of t only when the hash values of $f_{s,t}$ and $f_{s,t+L+i}$ collide more than $H/2$ times.

If the collected duplication candidates include ξ or more consecutive subsequences ($\xi = 10$ in our experiments, i.e., about 0.5 second), then these consecutive subsequences are considered as static scenes and discarded.

Note that this coarse matching stage also involves a distance threshold, R, but this distance threshold is different from that used in the existing works in that R is not used for making the final decision. The coarse matching step is used to eliminate unnecessary computations; therefore, when choosing R, we do not have to consider much about the tradeoff between robustness and distinctiveness; we should just guarantee that the duplication of t is a subset of $\{c_{t,1}, c_{t,2}, \ldots, c_{t,k}\}$. In fact, in practice, R is not necessarily explicitly assigned, and we will discuss this in more detail in the Experimental Results section.

At the end of this stage, each subsequence t is associated with a set of potential duplication candidates $\{c_{t,1}, c_{t,2}, \ldots, c_{t,k}\}$. The duplication verification step is then performed between t and these candidates.

3.3. Duplication Verification. For each pair of subsequence t and its duplication candidate $c_{t,i}$, we perform image registration between the corresponding frames to check whether the two frames contain the same objects and whether the shape and position of the corresponding objects happen to be identical. If so, the registration will yield zero-valued offset fields. However, it is not easy to stably obtain correct registration results for degraded images. As shown in Figure 1, even the lossy compression itself can result in substantial changes between a frame and its copy, which usually cause registration faults. To solve this problem, we propose to find the stable regions in the frames and rely on these regions more than the unstable areas during the registration procedure. We use a variant of Harris cornerness response proposed in [19] to measure the stability of the local structure of a pixel:

$$M = \frac{AB - C^2}{A + B}, \tag{2}$$

where $A = g * (\partial F/\partial x)$, $B = g * (\partial F/\partial y)$, and $C = g * ((\partial F/\partial x)(\partial F/\partial y))$; F denotes a given frame, g is a two-dimensional Gaussian kernel, and $*$ denotes the convolution operation.

For a frame F, a large value at $M(x, y)$ indicates that both eigenvalues of the autocorrelation matrix corresponding to $F(x, y)$ are large. This means that the signal changes significantly in two orthogonal directions; such points have been shown to be stable under various conditions except for scale change [20, 21]. We use M to weight different regions in a frame during the registration process, and the registration objective can be written as

$$E(\mathbf{v}) = \sum_{x,y} \mathbf{W}_D(x, y) \cdot D(\mathbf{v}(x, y)) + \sum_{\langle (m,n),(x,y)\rangle \in \mathcal{N}} \mathbf{W}_S(x, y) \cdot S(\mathbf{v}(x, y), \mathbf{v}(m, n)), \tag{3}$$

where D is the data term which measures the difference between the local structures around the matching pixels, S is the smoothness term which guarantees that neighbouring pixels have similar offsets, $\mathbf{v}(x, y)$ is the offset for point (x, y), and $\langle (m, n), (x, y)\rangle$ denotes an edge in the 4-neighbourhood system $\mathcal{N}$.

$\mathbf{W}_D$ are $\mathbf{W}_S$ are weighting matrices such that

$$W_D(x, y) = \begin{cases} 1 + M'(x, y), & \text{if } M'(x, y) > \varepsilon, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

$$W_S(x, y) = \max\left(W_D(x, y), \kappa\right), \tag{5}$$

where $M'(x, y) = \max_{-r \le a,b \le r}\{\widetilde{M}(x + a, y + b)\}$, with $\widetilde{M}$ being the normalized version of M. We use the maximal filtering to diffuse the impact of the stable points to a small range around them.

ε in (5) is quite a small value ($1e - 5$ in our implementation) used to mask out the excessively smooth regions when computing the data term (3). Although the Harris cornerness response weights the smooth regions less, the areas which are excessively flat still cause troubles during registration. The local structure of such regions can be easily changed by small perturbation. Figure 1(f) is an apparent illustration of this phenomenon, where large bright spots (significant differences between visual word indices) all locate on the wall or the floor, which are both rather smooth. Such excessively smooth regions will result in high data cost which is inconsistent with the real situation. Based on this observation, we set the threshold ϵ to remove the impact of the data costs in those areas. As a consequence, the offset field within those regions is completely controlled by the smoothness term (4); we therefore add a truncation value κ in (6) to guarantee that the smoothness constraint is always above a certain level.

The data term (3) and smoothness term (4) are, respectively, defined as

$$D(\mathbf{v}(x, y)) = \left\| f_{\text{src}}(x, y) - f_{\text{tar}}\left(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y)\right)\right\|_1, \tag{6}$$

$$S(\mathbf{v}(x, y), \mathbf{v}(m, n)) = \min\left(\alpha\left|\mathbf{v}_x(x, y) - \mathbf{v}_x(m, n)\right|, d\right) + \min\left(\alpha\left|\mathbf{v}_y(x, y) - \mathbf{v}_y(m, n)\right|, d\right), \tag{7}$$

where $f_{\text{src}}(x, y)$ is the feature for point (x, y) in the source frame, $f_{\text{tar}}(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y))$ denotes the feature for $(x + \mathbf{v}_x(x, y), y + \mathbf{v}_y(x, y))$ in the target frame (we use the SIFT descriptor [22] extracted on a single scale as the feature for each pixel), and $\mathbf{v}_x(x, y)$ and $\mathbf{v}_y(x, y)$ denote the offset in horizontal and vertical directions, respectively. We use the truncated l_1-norm in (8) to account for discontinuities in the offset field, and α is used for balancing the data term (3) and smoothness term (4) (α can also be combined into (5), we write it here for clearness).

We use the dual-layer loopy belief propagation in [23] to minimize the objective function $E(\mathbf{v})$. By decoupling the smoothness term in (4) into two parts in (8) corresponding to two directions, the complexity of message update in each iteration is reduced from $O(nL^4)$ to $O(nL^2)$, where n is the number of pixels in each frame and L is the number of possible offsets in each direction. The complexity is further reduced to $O(nL)$ by the distance transform proposed in [24]. The multigrid message passing scheme in [21] is also exploited to significantly reduce the total number of iterations.

Optical flow (e.g., [25]) or SIFT flow [23] can also be used for image registration, which is intrinsically a pixel-wise correspondence estimation process. However, neither of them can obtain expected results for degraded videos. The difference between our objective and that of optical flow and SIFT flow is obvious: we encode the stability information of different regions into the matching objective, which makes our method quite robust against video degradation. Moreover, in our objective, there is no small displacement term which is used in SIFT flow so that the registration can be more sensitive to subtle changes between two frames. Figure 3 shows two representative examples demonstrating the difference between the three methods. The offset fields are visualized with the color encoding scheme in [26]; please see Figure 4 for more details.

Compared with the typical methods for FDD [4–8], our method relies on image registration rather than the feature extraction and thresholding strategy. Conforming to the data similarity and smoothness constraints, the correspondences between the pixels of two frames are established in an "optimal" manner through a probabilistic inference process (i.e., the dual-layer loopy belief propagation); furthermore, the pixels with high Harris corner response are usually located on the boundaries of objects; therefore, when Harris corner response is integrated into the registration objective, the registration can be, to some extent, considered

(a) (b) (c) (d) (e)

Figure 3: The comparison between the three methods for image alignment. Top row: (a, b) a frame and its copy. Bottom row: (a, b) two consecutive frames. The video is recorded by using a still camera. The person in a black coat is walking toward left, and the person's gesture slightly changed across the two frames. (c)–(e) In both rows, the offset fields calculated based on optical flow [25] (we use the implementation in [27]), SIFT flow [23], and our method, respectively. Despite the impressive result for the motion of the walking person, optical flow is too sensitive to perturbations caused by lossy compression; it gets nonzero offsets all over the fields for both cases, particularly for the wall region in the first row. SIFT flow obtains better result than optical flow for the first case, but there is still a large area of nonzero values corresponding to the smooth wall. On the contrary, SIFT flow is not sensitive enough to subtle changes: it fails to detect the motion of the person in the second case. In contrast, our method correctly calculates both offset fields.

as an object-level matching process. As a result, even though the registration objective involves more parameters (i.e., κ, α, and d) than typical feature extraction and thresholding-based methods do, we will show in our experimental part that once the parameters are calibrated, the proposed method can perform more robustly than the feature extraction and thresholding-based methods.

4. Experimental Results

4.1. The Dataset. As far as we know, there is no publicly available dataset dedicated for FDD evaluation. Therefore, we created a dataset to evaluate the performance of the proposed method, especially for the degraded cases. We captured five indoor and eight outdoor video clips (named "v01" to "v13," and "v01"~"v05" are indoor scenes) with Panasonic HDC-Z10000GK camcorder. The videos were shot in the Science Park of Harbin Institute of Technology. The clips are captured from different scenes, and their contents include characters, landscapes, buildings, and plants. Several screenshots from our dataset are shown in Figure 5. The video clips are H.264 encoded by the built-in codec, and then we convert these clips into the .mp4 format with Adobe Premiere Pro CS 5.5. The resolution of the clips is 1920×1080, and the frame rate is 25 FPS. Based on these original clips, we created three forgery subsets: the MCOMP subset, the MCOMP + AGN subset, and the MCOMP + INT subset. The details of these subsets are listed in Table 1. Each original clip corresponds to 9 forged versions, and the whole dataset consists of 117 forged videos in all.

The magnitude of the additive Gaussian noise and intensity change is moderate so that they are hardly perceptible. The duration of the forged video clips varies from 8 seconds to 30 seconds.

4.2. The Efficiency of LSH-Based Coarse Matching. As mentioned above, given a subsequence t, in the coarse matching stage, we exploit LSH to find C, with C being the set of duplication candidates of t. In theory, to use the p-stable distribution-based LSH, we have to assign the distance threshold R and the error probability P_e to determine the parameter ω in (1). However, since we only use LSH as a coarse matcher and the result of the coarse matching does not have to be quite accurate, we should just make sure that $d \in C$, where d is the duplication of t. In this sense, we can determine ω by training rather than by firstly assigning R and P_e and then calculating ω from R and P_e.

To make sure $d \in C$, we define ϕ as the completeness of the correctly collected duplication candidates, and for the training set, we should choose ω such that

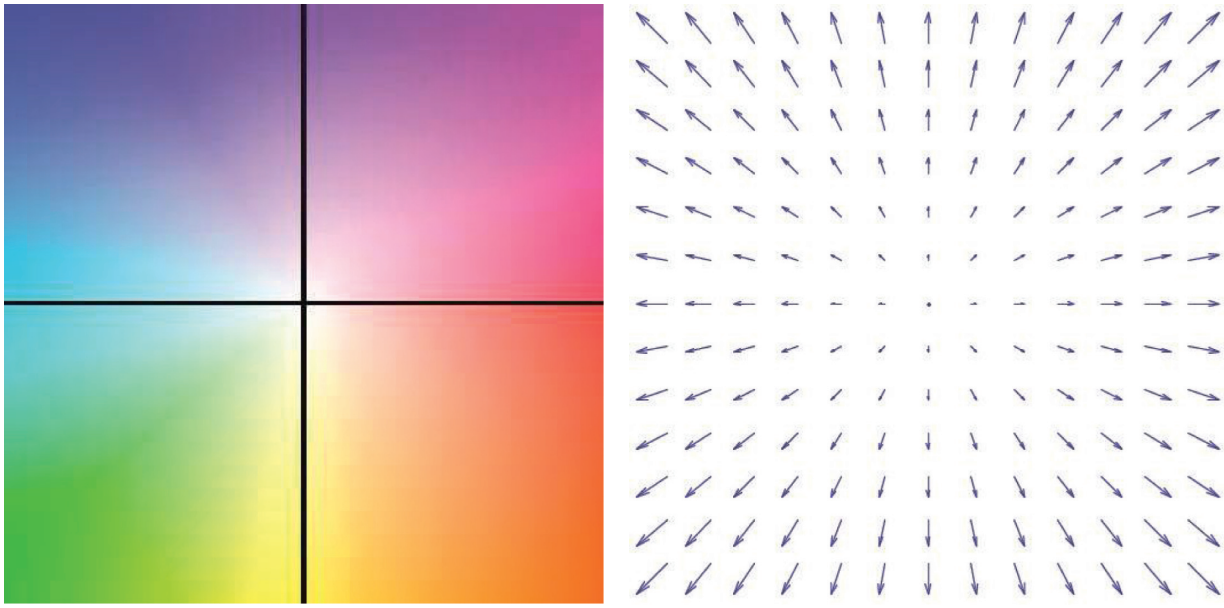
$$\phi = \frac{n_c}{n_g} = 1, \tag{8}$$

where n_c is the number of correctly collected duplication candidates and n_g is the actual number of duplicated subsequences. Given the premise in (9), we use the average number of collisions, n_{ave}, to measure the efficiency of the coarse matching step as follows:

$$n_{\text{ave}} = \frac{n_t}{T - L + 1}, \tag{9}$$

where n_t is the total number of collisions. It is straightforward that n_{ave} monotonically increases with ω, and we prefer smaller n_{ave} when (9) is guaranteed.

We randomly select four video clips which are most seriously degraded (i.e., have been attacked by additive Gaussian noise of the standard deviation of 10 or by downscaling the intensity by 3%) from the MCOMP + AGN and MCOMP + INT subset, respectively, to train the parameter ω. The block size of the block-wise GIST feature is 25×25, the subsequence length L is set to 5, and we scale each frame to 12.5% of their original size before feature extraction. We constructed 80 hash tables for the coarse matching stage; therefore, a pair of subsequences is considered to be identical only when their hash values collide more than 40 times. Under this configuration, ϕ and n_{ave} vary with ω as shown in Figure 6. We set $\omega = 0.45$, where ϕ reaches 1, and $n_{\text{ave}} \approx 0.094$.



(a) (b)

Figure 4: The color encoding of offset fields. (a) The orientation and magnitude of the offsets are represented by the hue and saturation of the corresponding pixel. (b) The sketch map illustrating the offset vectors relevant to different colors in (a).

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5: Sample pictures captured from the clips in our dataset. (a, i) Indoor scenes. (b)–(h) Outdoor scenes.

It is obvious that the value of ϕ and n_{ave} depends on the content of the videos. We list in Table 2 the mean values of ϕ and n_{ave} of the 9 forged versions corresponding to each original clip. Most of the mean values of ϕ are above 0.98. The two smaller values (0.94 for v03 and v13) are both caused by intensity degradation. For video clip v03 in group

Table 1: The description of the three forgery subsets.

Subset	Description
MCOMP	MCOMP stands for multiple compression. For each original clip, we use MATLAB R2014a to randomly copy k ($5 \le k \le 35$) consecutive frames and paste them to another random position in the timeline and resave the clips with the quality factors of 100, 80, and 60, respectively. Finally, each clip has been compressed three times (by camcorder, Adobe Premiere, and MATLAB, respectively). We denote the three levels of compression as MCOMP100, MCOMP80, and MCOMP60 groups, respectively.
MCOMP + AGN	The same steps as the MCOMP100 group are performed. Besides, before resaving the forged clips, additive Gaussian noise with the standard deviations of 1, 5, and 10 is added to the target frames, i.e., each original clip corresponds to three forged clips subject to different levels of additive noise. For simplicity, in the rest of this paper, we refer to these three levels of forgeries as AGN1, AGN5, and AGN10 groups, respectively.
MCOMP + INT	The same steps as the MCOMP100 group are performed. Besides, before resaving the forged clips, the intensity of the pixels in the target frames is downscaled to 99%, 97%, and 95% of their original values. In the following, we refer to these three levels of forgeries as INT99, INT97, and INT95 groups, respectively.

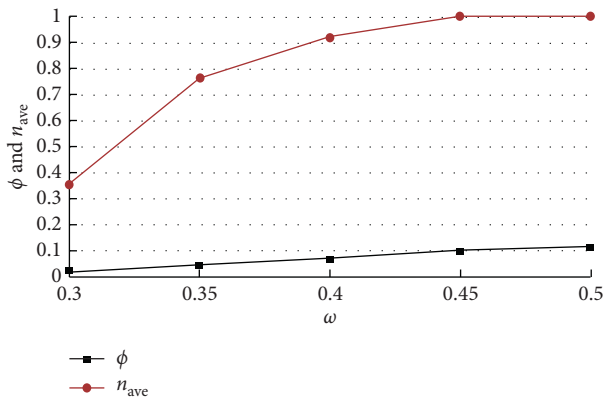


Figure 6: ϕ and n_{ave} increase as ω gets larger. ϕ reaches 1 when $w = 0.45$.

INT95, the value of ϕ was only 0.61. This is because the intensity of the pixels of the duplicated frames is large; therefore, a small scaling factor can result in remarkable intensity change. On average, 2% duplicated subsequences are missed during the coarse matching stage.

On the contrary, the average of $\text{mean}_{n_{ave}}$ for different scenes is 0.10; this implies that, in our duplication verification stage, on average, we just need to perform 0.1 comparison for each subsequence. In contrast, without the coarse matching stage, on average, we have to compare each subsequence with about $T/2$ other subsequences, where T is the number of frames. T is typically larger than 200 in our dataset and can be much larger in practice. In this sense, for our dataset, the computational load for the duplication verification stage is reduced by 3 orders of magnitude. We will show later that it is worth performing this coarse matching step, in spite of its $O(T \cdot (T+1)/2)$ time complexity.

4.3. The Detection Capability. In this section, we investigate the detection capability of our method. In our implementation, we randomly select 7 forged clips to calibrate κ, α, and d, and κ, α, and d are empirically set to be 1.8, 635, and 12,800, respectively. The frames are resized by a factor of 25% (for acceleration) before registration. We binarized the resulted offset field images and masked out the nonzero regions whose area is less than 0.1% of the whole field image to account for outliers. We use precision (10), recall (11), and F_1-score (12) to evaluate the performance:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{10}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{11}$$

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \tag{12}$$

where TP, FP, and FN are the number of correctly detected duplication frame pairs, the number of falsely detected duplication frame pairs, and the number of undetected duplication pairs, respectively.

We compared our method with [4, 8] (denoted Farid and Li, respectively), and the related parameters are set to be identical to those in [4, 8], respectively. The comparison results for the MCOMP subset are shown in Tables 3–5. The label "v01" in the first row denotes the forged version of video clip v01 in the current forgery group, and "average" means the average value of v01 to v13, and the same hereinafter. For v07 to v12 in the MCOMP100 group, all three methods performed perfectly. For v05 and v13, our method obtained rather low precision; this is because some shots in v05 and v13 are almost still, and the duplication verification step failed to differentiate the excessively similar frames. On the contrary, [4] is quite effective in terms of discriminative power. Our method outperformed the other two in the first four cases. It should be noted that [8] detected none of the duplicated frames in v02 to v05 and v13 for differing reasons. In [8], when the temporal correlations of the frames in a subsequence are all above a certain value, this subsequence is considered as static and then discarded; therefore, in v05 and v13, the subsequences whose frames are quite similar to each other have not been compared with other subsequences at all. In contrast, the duplicated frames in v02 to v04 are missed due to the inappropriate correlation threshold.

For the MCOMP80 group, the precision of [4] for v05 and v13 dropped to 0.44 and 0.50, respectively, and the performance of [4] for the rest of the clips just slightly changed. The performance of [8] was rather poor for this

Table 2: The mean values of ϕ and n_{ave} of the forged versions corresponding to each original clip.

	v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Mean_{ϕ}	0.99	0.98	0.94	1.00	0.99	1.00	0.96	0.99	0.98	1.00	1.00	1.00	0.94	0.98
$\text{Mean}_{n_{ave}}$	0.38	0.04	0.08	0.10	0.34	0.10	0.03	0.06	0.03	0.05	0.02	0.02	0.07	0.10

Table 3: The comparison results for the MCOMP100 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.22	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.21	0.88
	Farid [8]	1.00	0.00	0.00	0.00	0.00	0.74	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.60
	Li [4]	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.85
Recall	Ours	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.88	0.99
	Farid [8]	0.17	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.30	0.70	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.77
F_1-score	Ours	0.98	1.00	1.00	1.00	0.36	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.34	0.90
	Farid [8]	0.29	0.00	0.00	0.00	0.00	0.85	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.46	0.82	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79

Table 4: The comparison results for the MCOMP80 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.12	1.00	1.00	0.50	1.00	1.00	1.00	1.00	0.09	0.82
	Farid [8]	1.00	0.00	0.00	0.00	0.00	0.82	1.00	0.00	1.00	0.54	0.00	0.00	0.00	0.34
	Li [4]	0.00	0.00	1.00	1.00	0.44	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.50	0.76
Recall	Ours	0.97	1.00	1.00	1.00	1.00	1.00	0.83	1.00	1.00	1.00	1.00	1.00	1.00	0.98
	Farid [8]	0.38	0.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	0.91	0.00	0.00	0.00	0.33
	Li [4]	0.00	0.00	1.00	0.87	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.83
F_1-score	Ours	0.98	1.00	1.00	1.00	0.22	1.00	0.91	0.67	1.00	1.00	1.00	1.00	0.17	0.84
	Farid [8]	0.55	0.00	0.00	0.00	0.00	0.90	1.00	0.00	1.00	0.68	0.00	0.00	0.00	0.32
	Li [4]	0.00	0.00	1.00	0.93	0.61	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.65	0.78

Table 5: The comparison results for the MCOMP60 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.16	1.00	1.00	1.00	1.00	1.00	0.50	1.00	0.37	0.85
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	1.00	0.54
Recall	Ours	0.87	1.00	1.00	1.00	1.00	1.00	1.00	0.75	0.83	1.00	0.50	1.00	0.96	0.92
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	0.48	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.96	0.50
F_1-score	Ours	0.93	1.00	1.00	1.00	0.28	1.00	1.00	0.86	0.91	1.00	0.50	1.00	0.53	0.85
	Farid [8]	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	0.38
	Li [4]	0.00	0.00	0.65	0.00	1.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.98	0.51

group: it failed to detect 8 out of 13 forged video clips. In this group, the precision for v08 and recall for v07 of our method decreased to 0.50 and 0.83, respectively, and the results for other clips were almost the same with those in the MCOMP100 group.

When the quality factor of the last compression dropped to 60, the performance of [4] significantly decreased. Specifically, 6 video clips were mistakenly judged as unchanged. In contrast, most results of our method maintained a relatively stable value.

The detection results for the MCOMP + AGN subset are shown in Tables 6–8. In the AGN1 group (Table 6), our method outperformed the other two for the first four cases. Our recall rate for v08 is a little lower than the other two. Besides, for v05 and v13, [4] still obtained better results than our method, while the precision for v05 dramatically dropped to less than 0.50. [8] obtained the worst results for v01 to v03, v05, v06, and v13. In Table 7, it is interesting that the precision of [4] for v05 in the AGN5 group is 1.00, which was supposed to be less than 0.50 due to the stronger noise. According to our observation, the difference between the source and target frames in v05 in this group is indeed small, which may be caused by the encoding mechanism of the lossy compression. In Table 8, for most cases, the precision and recall of [4, 8] dropped to 0. Contrarily, our method is

Table 6: The comparison results for the AGN1 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.07	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.17	0.87
	Farid [8]	0.00	0.00	0.00	1.00	0.00	0.62	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.59
	Li [4]	0.00	0.00	1.00	1.00	0.46	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.80
Recall	Ours	0.92	1.00	1.00	1.00	1.00	1.00	1.00	0.97	1.00	1.00	1.00	1.00	1.00	0.99
	Farid [8]	0.00	0.00	0.00	0.56	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79	0.00	0.57
	Li [4]	0.00	0.00	0.86	0.53	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.80
F_1-score	Ours	0.96	1.00	1.00	1.00	0.14	1.00	1.00	0.98	1.00	1.00	1.00	1.00	0.29	0.87
	Farid [8]	0.00	0.00	0.00	0.72	0.00	0.76	1.00	1.00	1.00	1.00	1.00	0.88	0.00	0.57
	Li [4]	0.00	0.00	0.92	0.69	0.63	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.79

Table 7: The comparison results for the AGN5 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.22	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.12	0.87
	Farid [8]	1.00	0.00	0.00	0.00	0.00	1.00	1.00	0.37	0.71	0.59	1.00	1.00	0.00	0.51
	Li [4]	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.85
Recall	Ours	1.00	1.00	1.00	1.00	0.97	1.00	0.97	0.88	1.00	1.00	1.00	1.00	0.86	0.98
	Farid [8]	0.80	0.00	0.00	0.00	0.00	1.00	0.74	1.00	1.00	0.76	1.00	0.22	0.00	0.50
	Li [4]	0.00	0.00	1.00	0.35	1.00	1.00	0.35	1.00	1.00	1.00	1.00	1.00	1.00	0.75
F_1-score	Ours	1.00	1.00	1.00	1.00	0.36	1.00	0.99	0.94	1.00	1.00	1.00	1.00	0.22	0.88
	Farid [8]	0.89	0.00	0.00	0.00	0.00	1.00	0.85	0.54	0.83	0.66	1.00	0.36	0.00	0.47
	Li [4]	0.00	0.00	1.00	0.52	1.00	1.00	0.52	1.00	1.00	1.00	1.00	1.00	1.00	0.77

Table 8: The comparison results for the AGN10 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.08	0.86
	Farid [8]	1.00	0.00	0.00	1.00	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.54
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00	1.00	0.00	0.23
Recall	Ours	1.00	1.00	1.00	1.00	1.00	1.00	0.70	1.00	1.00	1.00	1.00	1.00	0.88	0.97
	Farid [8]	0.94	0.00	0.00	0.33	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.48
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.83	0.00	0.00	1.00	1.00	0.00	0.22
F_1-score	Ours	1.00	1.00	1.00	1.00	0.18	1.00	0.82	1.00	1.00	1.00	1.00	1.00	0.15	0.86
	Farid [8]	0.97	0.00	0.00	0.50	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	0.00	0.50
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.91	0.00	0.00	1.00	1.00	0.00	0.22

rather stable across different groups, and the performance hardly changed.

The detection results for the MCOMP + INT subset are shown in Tables 9–11. Except for v05 and v13, our method is on par with or excelled the other two methods for most cases. Since [8] used correlation of pixel intensity as their feature, which is rather robust against intensity change, in this subset, [8] performed better than it did in the COMP + AGN subset. On the contrary, [4] is rather sensitive to intensity change. For the INT95 group, [4] even detected none duplicated frames at all for 10 out of 13 forged clips.

When comparing frames in degraded videos, the distances between the features can easily fall out of the range of a fixed threshold. As can be seen from Tables 3–11, when the degradation gets stronger, more than half of the forged video clips have been falsely judged as innocent by [4, 8]: none of the duplicated frames has been detected. By contrast, our method performed stably across different test groups. Although, occasionally, our method performed worse than the other two methods (especially for v05 and v13), the average of the precision, recall, and F_1-score, without exception, significantly outperformed those of [4, 8]. Even for the strongest degradation groups, i.e., MCOMP60, MCOMP + AGN10, and MCOMP + INT95, the average values of precision, recall, and F_1-score of our method are above 0.8 (for the worst case, $F_1 = 0.81$ in INT95).

4.4. The Running Time. The running time of the three methods is closely related to the content of the videos. When comparing subsequences, once any corresponding pair of frames is found to be not identical, the comparison between the current pair of subsequences terminates. Therefore, the video clips whose frames are highly similar to each other will result in more processing time. The comparison between the running time of the three methods is given in Table 12. The frames are scaled by a factor of 25% for all the three methods, and all the experiments are conducted on a workstation with

Table 9: The comparison results for the INT99 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.09	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.09	0.86
	Farid [8]	1.00	0.00	1.00	1.00	0.00	0.66	1.00	1.00	1.00	0.00	1.00	1.00	0.00	0.67
	Li [4]	0.00	0.00	1.00	0.00	0.34	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.72
Recall	Ours	0.94	0.83	0.83	1.00	1.00	1.00	0.93	1.00	1.00	1.00	1.00	1.00	0.89	0.95
	Farid [8]	0.16	0.00	0.83	1.00	0.00	0.94	0.75	0.36	1.00	0.00	1.00	0.52	0.00	0.50
	Li [4]	0.00	0.00	0.97	0.00	1.00	1.00	0.71	1.00	1.00	1.00	1.00	1.00	1.00	0.74
F_1-score	Ours	0.97	0.91	0.91	1.00	0.17	1.00	0.96	1.00	1.00	1.00	1.00	1.00	0.17	0.85
	Farid [8]	0.28	0.00	0.91	1.00	0.00	0.78	0.86	0.53	1.00	0.00	1.00	0.68	0.00	0.54
	Li [4]	0.00	0.00	0.98	0.00	0.51	1.00	0.83	1.00	1.00	1.00	1.00	1.00	1.00	0.72

Table 10: The comparison results for the INT97 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.19	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.15	0.87
	Farid [8]	0.00	0.00	0.00	1.00	0.00	0.35	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.57
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.69
Recall	Ours	0.83	1.00	1.00	1.00	0.96	1.00	1.00	0.82	1.00	1.00	1.00	1.00	0.96	0.97
	Farid [8]	0.00	0.00	0.00	0.32	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95	0.00	0.56
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	0.45	1.00	1.00	1.00	1.00	0.83	0.64
F_1-score	Ours	0.91	1.00	1.00	1.00	0.32	1.00	1.00	0.90	1.00	1.00	1.00	1.00	0.26	0.88
	Farid [8]	0.00	0.00	0.00	0.48	0.00	0.52	1.00	1.00	1.00	1.00	1.00	0.96	0.00	0.54
	Li [4]	0.00	0.00	0.00	0.00	1.00	1.00	1.00	0.63	1.00	1.00	1.00	1.00	0.91	0.66

Table 11: The comparison results for the INT95 group.

		v01	v02	v03	v04	v05	v06	v07	v08	v09	v10	v11	v12	v13	Average
Precision	Ours	1.00	1.00	1.00	1.00	0.16	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.25	0.88
	Farid [8]	0.00	0.00	0.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.62
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	1.00	1.00	0.00	0.23
Recall	Ours	0.54	1.00	0.61	1.00	1.00	1.00	1.00	0.23	1.00	1.00	1.00	1.00	0.86	0.86
	Farid [8]	0.00	0.00	0.00	0.41	0.00	1.00	1.00	1.00	1.00	0.73	1.00	1.00	0.00	0.55
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.00	0.00	0.00	1.00	0.97	0.00	0.18
F_1-score	Ours	0.70	1.00	0.76	1.00	0.27	1.00	1.00	0.38	1.00	1.00	1.00	1.00	0.38	0.81
	Farid [8]	0.00	0.00	0.00	0.58	0.00	1.00	1.00	1.00	1.00	0.84	1.00	1.00	0.00	0.57
	Li [4]	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	1.00	0.99	0.00	0.19

Table 12: The averaged running time (seconds) of each scene of the three methods.

Scenes	Average duration (seconds)	Average running time		
		Ours	Farid [8]	Li [4]
v01	10	447	**438**	498
v02	11	**214**	996	567
v03	8	**195**	228	324
v04	8	218	**162**	375
v05	17	1979	**150**	1789
v06	9	**263**	535	450
v07	19	**383**	752	1900
v08	9	**226**	438	443
v09	13	**258**	394	860
v10	14	**293**	655	1010
v11	28	**461**	5226	3932
v12	30	548	**242**	4732
v13	29	951	**169**	4020

Table 13: The average running time (seconds) of each step in our method.

Steps	Average running time
Frame extraction	32
Coarse matching	209
Duplication verification	252

Intel Core i7-2600 processor and 24 GB RAM. We implemented the three methods with MATLAB R2014a.

As mentioned earlier, in [8], if the correlation coefficients between the frames within a given subsequence are all above a predefined value, the subsequence will be considered as static and discarded. Such subsequences will not be compared to other subsequences, and that is why the running time of [8] is so short for scenes v05 and v13 (recall that [8] detected none of the forgeries corresponding to these two scenes).

In fact, both the structural similarity and correlation coefficients used in [4, 8], respectively, can be computed much faster than the image registration process in our method. For each single pair of frames to be compared, the structural similarity and correlation coefficients can be calculated in about 0.03 and 0.05 second, respectively. In contrast, in our method, the image registration procedure takes about 4 seconds for each pair of frames to be compared. Even so, our method is faster than the other two for most cases. The coarse matching step plays an important role for acceleration. As we have demonstrated in Table 2, the total number of subsequences which need finer duplication verification can be reduced by several orders of magnitude, especially for the video clips whose content changes rapidly across frames. The average running time of each step in our method is listed in Table 13.

The coarse matching step accounts for about 42% of the total running time, and it takes less than one second for each frame on average. Such a step is well worth performing: without coarse matching, even a 10-second video clip will cost us several hours to detect the forgery.

5. Conclusion and Future Work

In this paper, we proposed a new method for frame duplication detection, particularly for the degraded videos. Our method detects duplication forgeries in a coarse-to-fine manner and consists of two steps: coarse matching and duplication verification. In the coarse matching stage, we use locality-sensitive hashing to precluster the visually similar subsequences. Through coarse matching, the total number of subsequences which need finer duplication verification can be reduced by several orders of magnitude. The duplication verification step exploits image registration to identify the identical subsequences. We encode the stability information of different regions into the registration objective function such that the registration can work stably for degraded videos. Being different from existing methods, our detection process does not rely on a fixed distance threshold, which is typically unreliable for degraded videos. Experimental results show that our method outperformed state-of-the-art methods for most cases and exhibited outstanding robustness under different conditions. However, our method cannot distinguish between highly similar frames; as a result, for the video clips whose content just slightly changes across frames, the precision can be rather low. Further efforts should be made to improve the discriminative power of the registration process.

Data Availability

The dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61771168) and the Natural Science Foundation of Heilongjiang Province of China (Grant no. F2017014).

References

[1] Z. Huang, L. Fang, and S. Li, "Subpixel-pixel-superpixel guided fusion for hyperspectral anomaly detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 99, pp. 1–10, 2020.

[2] D. Meng, X. Wang, M. Huang, L. Wan, and B. Zhang, "Robust weighted subspace fitting for DOA estimation via block sparse recovery," *IEEE Communications Letters*, vol. 24, no. 3, pp. 563–567, 2020.

[3] Y. Wang, L. Wang, C. Yu et al., "Constrained-target band selection for multiple-target detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 8, pp. 6079–6103, 2019.

[4] F. Li and T. Huang, "Video copy-move forgery detection and localization based on structural similarity," in *Proceedings of the 3rd International Conference on Multimedia Technology (ICMT 2013)*, London, UK, June 2014.

[5] S. Y. Liao and T.-Q. Huang, "Video copy-move forgery detection and localization based on tamura texture features," in *Proceedings of the 2013 6th International Congress on Image and Signal Processing (CISP)*, Hangzhou, China, December 2013.

[6] G. S. Lin, J.-F. Chang, and C.-H. Chuang, "Detecting frame duplication based on spatial and temporal analyses," in *Proceedings of the 2011 6th International Conference on Computer Science & Education (ICCSE)*, Singapore, August 2011.

[7] D. Tralic, S. Grgic, and B. Zovko-Cihlar, "Video frame copy-move forgery detection based on cellular automata and local binary patterns," in *Proceedings of the 2014 X International Symposium on Telecommunications (BIHTEL)*, Sarajevo, Bosnia, October 2014.

[8] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting duplication," in *Proceedings of the 9th Workshop on Multimedia & Security*, pp. 35–42, Cardiff, Wales, UK, July 2007.

[9] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[10] A. Subramanyam and S. Emmanuel, "Video forgery detection using hog features and compression properties," in *Proceedings of the 2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 89–94, Banff, AB, Canada, September 2012.

[11] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy retrieval using distortion-based probabilistic similarity search," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 293–306, 2007.

[12] M. Douze, H. Jegou, and C. Schmid, "An image-based approach to video copy detection with spatio-temporal post-filtering," *IEEE Transactions on Multimedia*, vol. 12, no. 4, pp. 257–266, 2010.

[13] D. Q. Zhang and S.-F. Chang, "Detecting image near-duplicate by stochastic attributed relational graph matching with learning," in *Proceedings of the 12th Annual ACM international Conference on Multimedia*, pp. 877–884, New York, NY, USA, October 2004.

[14] O. Chum, J. Philbin, M. Isard, and A. Zisserman, "Scalable near identical image and shot detection," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, pp. 549–556, Amsterdam, The Netherlands, 2007.

[15] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, and P. Moulin, "Robust image hashing," in *Proceedings of the 2000 International Conference on Image Processing*, pp. 664–666, Vancouver, Canada, 2000.

[16] S. S. Kozat, R. Venkatesan, and M. K. Mihc¸ak, "Robust perceptual image hashing via matrix invariants," in *Proceedings of the 2004 International Conference on Image Processing (ICIP'04)*, pp. 3443–3446, Singapore, October 2004.

[17] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[18] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on *p*-stable distributions," in *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pp. 253–262, Brooklyn, NY, USA, 2004.

[19] J. A. Noble, "Finding corners," *Image and Vision Computing*, vol. 6, no. 2, pp. 121–128, 1988.

[20] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference*, Manchester, UK, September 1988.

[21] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, 2000.

[22] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pp. 1150–1157, Kerkyra, Greece, September 1999.

[23] C. Liu, J. Yuen, and A. Torralba, "Sift flow: dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, 2011.

[24] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.

[25] A. Bruhn, J. Weickert, and C. Schnorr, "Lucas/kanade meets horn/schunck: combining local and global optic flow methods," *International Journal of Computer Vision*, vol. 61, no. 3, pp. 211–231, 2005.

[26] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.

[27] C. Liu, *Beyond pixels: exploring new representations and applications for motion analysis*, Ph.D. dissertation, 2009.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6637936, 11 pages
https://doi.org/10.1155/2021/6637936

Research Article

Attention-Guided Digital Adversarial Patches on Visual Detection

Dapeng Lang,[1] **Deyun Chen**,[1] **Ran Shi**,[2] **and Yongjun He**[1]

[1]*School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150001, China*
[2]*School of Computer Science and Technology, Heilongjiang University, Harbin 150001, China*

Correspondence should be addressed to Deyun Chen; chendeyun@hrbust.edu.cn

Received 18 December 2020; Revised 25 January 2021; Accepted 19 March 2021; Published 8 April 2021

Academic Editor: Tom Chen

Deep learning has been widely used in the field of image classification and image recognition and achieved positive practical results. However, in recent years, a number of studies have found that the accuracy of deep learning model based on classification greatly drops when making only subtle changes to the original examples, thus realizing the attack on the deep learning model. The main methods are as follows: adjust the pixels of attack examples invisible to human eyes and induce deep learning model to make the wrong classification; by adding an adversarial patch on the detection target, guide and deceive the classification model to make it misclassification. Therefore, these methods have strong randomness and are of very limited use in practical application. Different from the previous perturbation to traffic signs, our paper proposes a method that is able to successfully hide and misclassify vehicles in complex contexts. This method takes into account the complex real scenarios and can perturb with the pictures taken by a camera and mobile phone so that the detector based on deep learning model cannot detect the vehicle or misclassification. In order to improve the robustness, the position and size of the adversarial patch are adjusted according to different detection models by introducing the attachment mechanism. Through the test of different detectors, the patch generated in the single target detection algorithm can also attack other detectors and do well in transferability. Based on the experimental part of this paper, the proposed algorithm is able to significantly lower the accuracy of the detector. Affected by the real world, such as distance, light, angles, resolution, etc., the false classification of the target is realized by reducing the confidence level and background of the target, which greatly perturbs the detection results of the target detector. In COCO Dataset 2017, it reveals that the success rate of this algorithm reaches 88.7%.

1. Introduction

The development of DNN has yielded significant results in several fields. From image processing and self-driving cars to language processing and artificial intelligence, DNN is profoundly changing the role of computers in human production and life. DNN has been able to outperform humans in many fields [1–3]. Through the analysis of DNN structure, its performance is mainly achieved by using statistical learning on a large number of data to obtain an effective representation of the input space and extract advanced features from the original sensory data. In recent years, researchers have found that DNN networks are very sensitive to disturbance resistance. Even very small data changes, such as modifying one or more pixels in an image that is invisible to the human eye, can cause DNN to misclassify or even fail to detect the target. These images with small perturbations are called adversarial examples. Initially, the researchers thought the adversarial examples were due to the nonlinearity of the deep learning model. But then Goodfellow proved through experiments that the reason was the linear behaviour of the deep neural network in high dimensional space. At the same time, it can be proved that there is no essential difference between adversarial examples and natural examples, and the examples misclassified during model testing can be used as effective adversarial examples.

In the field of image recognition [2, 3, 4–10], the CNN model learns the features of the target to be detected by learning a large number of target image models and extracting advanced features from the image. In this process, the model not only learns the distribution of features but also learns the features of the graphic background and edge. In

general, the recognition accuracy will not affect the normal use of CNN. However, in the field of battlefield target recognition, vehicle identification, and security, attacks on algorithm models may have disastrous consequences. Such threats have attracted the attention of the research community and industry. Meanwhile, the method of generating adversarial examples is open and effective, such as Fast Gradient Sign Method (FGSM) [11], Projected Gradient Descent [12], DeepFool [13], etc. L-BFGS [3], as an optimization algorithm, is used in many other algorithms to generate adversarial samples. Another broad approach is to mislead the classifier by modifying a very small number of pixels in the image. There are also methods to add a patch in the image, perturbation classifier operation. This kind of patch is usually specially designed and generated. Experiments show that it is not effective to mask image features with just noise.

In practical application, there is generally no condition to modify or replace the whole image to be recognized. At the same time, the existing detector is very sensitive to the environment, angle, and clarity, and its attack effect is unstable in the experimental environment and real environment, so it is difficult to popularize in the real environment. The method of adding patch image on the image is more practical because the patch itself can be composed of various images and colours. It is not easy to attract attention, but the problem of the adversarial patch in practical application is also obvious. First of all, due to the influence of light and angle, the robustness of the patch needs to be strengthened urgently. After stretching, the patch is easy to lose its attack effect. In addition, the generation of patches depends on the algorithm structure of detection targets and detectors, so the transferability is not good. At present, the research mainly focuses on fixed small targets, such as the use of Stop signs. In this paper, electronic adversarial patches on the image are studied. In combination with the size and position of the specific target in the image, a proportional "adversarial patch" is generated, which is attached to the surface of the detection target to deceive and mislead the detector to detect the image. Figure 1 illustrates the effect.

2. Related Work

2.1. Digital Adversarial Examples. The software architecture and model structure in the white-box setting is visible to attackers, so it is easy to construct adversarial examples against classifiers. In general, through the analysis of the white-box settings, the need to be based are based on the following considerations. When attacking the photos and images of real road vehicles and cars, attackers can often obtain the detection model and algorithm structure by social engineering and reverse engineering so as to construct the adversarial examples conveniently. On the other hand, for the defence side, the analysis of attack means based on white-box setting is more conducive to the study of defence methods. In recent years, more and more researchers have applied the method of generating adversarial examples [13, 14] based on white-box setting to verify the transferability [12, 15] of black adversarial examples. By improving the white-box attack method, the migration of counter samples is better, and the black-box attack is carried out on other black-box models, which improves the effect of black-box attack. Attacks against physical objects are also on the rise [3, 11, 16–19].

Goodfellow proposed a fast gradient algorithm, which proved the existence of adversarial examples. Through the first-order approximation of the loss function [20–22], the false classification of the targets in the image was realized. In addition, the adversarial example generation method based on optimization ideas can achieve targeted attack by adding carefully chosen perturbations. The common point of these two methods is that they adopt the DAE method, namely digital adversarial examples. In the research of DAE, the current research direction mainly focuses on the detection of targets with small intraclass differences and large interclass differences, such as road STOP signs and human faces. The physical perturbations against physical targets have gradually become a research hotspot.

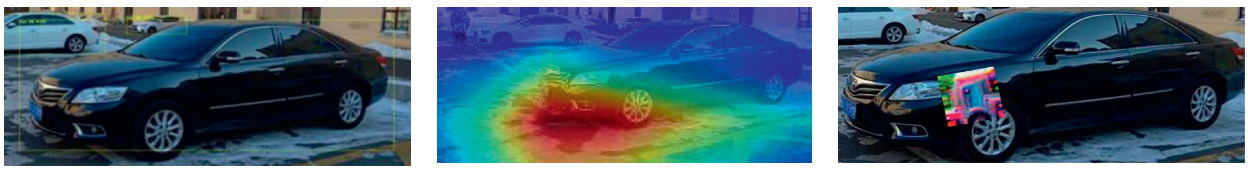
2.2. Object Detection. The experiment in this paper is mainly based on the target detector of YOLOv2 [6]. Its structure is based on convolution and pooling (downsampling). Finally, two fully connected layers are added. The input image is divided into $S \times S$ grids. If the coordinates of the center of the ground truth of an object fall into a grid, the grid is responsible for detecting the object. Each grid predicts B bounding boxes and their confidence scores and C category probabilities. The bounding box information (x, y, w, h) is the offset of the center position of the object relative to the grid position and the width and height, all of which are normalized. The confidence scores reflect whether the object is included and the accuracy of the position in the case of including the object, which is defined as follows:

$$\Pr(\text{Obj}) \times \text{IOU}_{\text{pred}}^{\text{truth}}, \quad \Pr(\text{Obj}) \in \{0, 1\}, \tag{1}$$

The optimization direction is to have the same prediction confidence as the ground truth IOU. The predicted conditional probability value of each grid cell is $C(\Pr(\text{Class}_i|\text{Obj}))$. In the test, each box obtains the specific category confidence by multiplying the category probability and the box confidence:

$$\Pr(\text{Class}_i|\text{Obj}) * \Pr(\text{Obj}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) \times \text{IOU}_{\text{pred}}^{\text{truth}}. \tag{2}$$

In order to train the detection network, the classification network is changed to a detection network, the last convolutional layer of the original network is removed, multiple convolutional layers (usually 1024 filters) are added, and each convolutional layer is followed by a 1×1 Convolutional layer, the number of outputs is the number required for detection. Finally, the YOLOv2 framework obtains the target class score through optimization of the cross-entropy loss function. Figure 2 shows the architecture of this process.



(a) (b) (c)

Figure 1: By detecting the key feature aggregation field in the image, the location and size of the generated adverse patch in this paper are located to achieve the purpose of the invisible detector. (a) The original car which can be detected; (b) visualize the field with high feature density; (c) the detector cannot detect the car with a patch.

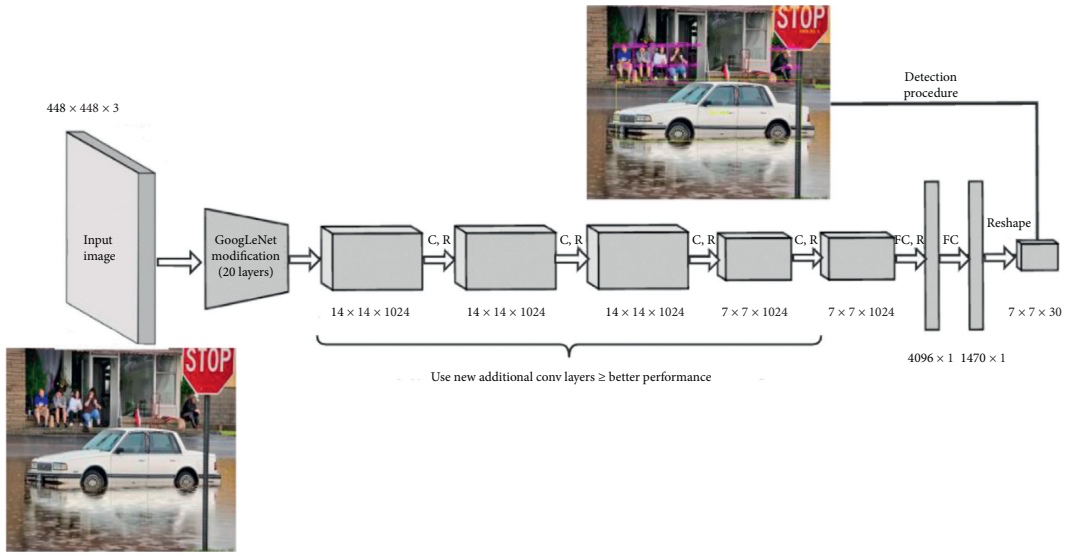


Figure 2: The typical way of detecting objects by YOLOv2.

2.3. Explainability of Attention. When adversarial patches are used as an attack method, studies have shown that the size, deformation, and illumination of the patch have a significant impact on the attack effect. However, the existing method mainly uses the random gradient descent method to generate and adjust the patch, and the effect is not stable during the application process, and the efficiency is not high. Our paper studies the formation principle of the attention mechanism [23, 24] through the method of guiding by the attention mechanism and uses the advantages of the attention mechanism from the perspective of the interpretability of the target detection algorithm. In this way, we can explore the inner workings of the deep learning framework. Attention is a mechanism used to improve the effectiveness of the Encoder + Decoder model based on RNN (LSTM or GRU). Due to the ability to distinguish attention mechanism, it is widely used in many fields such as machine translation, speech recognition, and image annotation.

In the detection process of the target detector, there is often a selective acquisition of some important parts of the observed object. Attention Mechanism can help the model assign different weights to each part of the input X, extract more critical and important information, and enable the model to make more accurate judgments without incurring greater costs for the calculation and storage. This is also the reason why the attention mechanism is so widely used. In order to fully analyse and understand the operating principle of the recognizer and provides a basis for generating adversarial patches. In this paper, the network structure is visualized in the way of class activation map visualization [25–27]. Through the heatmap, we can understand which features of the image play a key role in the image classification problem and locate the position of the object in the image.

2.4. Adversarial Patch. The adversarial patch is developed from the adversarial examples. The method of adding perturbance to the pixels of the original images makes the deep learning model like convolution neural network reduce the accuracy significantly. However, the adversarial patch does not consider whether it is perceptible to the human eyes and directly covers the "patch" on the detected target, which makes the detector misclassified or unrecognized in the detecting process [1, 18, 28, 29].

The mathematic definition would be given a patch patch, an image $x \in$ ImagData, where ImagData is an image dataset, targeted classification targetClass, the location of the patch location $\in L$, the set of the transformation of images trans $\in T$, and all the space information for the rotating, scaling, etc., all of which composed of an image operator OP (patch, x, location, trans). The generated patch is

updated iteratively by optimizing the objective function. The objective function is as follows:

$$\widehat{\text{patch}} = \underset{\text{Patch}}{\text{argmax}} E_{x\sim \text{Im}ag\text{Data},\text{trans}\sim T,\text{location}\sim L}\left[\log \Pr\left(\widehat{\text{patch}}|\text{OP}\left(\text{patch}, x, \text{location}, \text{trans}\right)\right)\right]. \tag{3}$$

Different from physical target hiding, the digital adversarial patch does not need to consider the dynamic environment information such as change of illumination and sudden deformation but generates the optimal adversarial structure according to the real-time snapshot. In the formula above, the image is transformed to improve the robustness of the adversarial patch so that the patch can still be effective under the operation of stretching and moving so as to complete the optimization of parameter expectation.

The adversarial patches generated in different works of the literature are shown in Figure 3 below:

3. Methods

3.1. Brief Overview of the AGAP. The goal of this paper is to generate an adversarial patch efficiently and accurately. By analysing the white-box setting of the target detection model, the algorithm generates the adversarial patch that can cheat the model detector. Based on the processing of bounding box information from the YOLO detector, the distribution of the key features of the target is analysed to form a heatmap, position, and colour based on heatmap. We attach these patches to the object that needs to be hidden in the image to hide the object. The core idea is to optimize the image at the pixel level. Based on the huge dataset, the detection score of the target detector is reduced. Our algorithm is attention-guided adversarial patch generation which is abbreviated by AGAP. The pipeline graph of our paper is in Figure 4 as follows.

3.2. Attention-Guided Feature Visualization. It is not necessary to consider whether the patch is perceptible to human eyes. The purpose of this paper is to solve the problem that the adversarial patches with a field of less than 10% are attached to images and pictures under the condition of a black-box detector, which may cause false classification or achieve unrecognized attack effect. This paper focuses on solving the two major problems in the generation of adversarial patches. The first is to construct and optimize the loss function in the process of generating the adversarial patches, which have robustness. On the other hand, we should pay attention to the location, the size, and the rotation angle of the patch. It is proved that the same patch is sensitive to location, light, size, and other factors. Even if the classifiers can be successfully attacked by patches digitally, they often do not work when they are printed or projected. Therefore, this paper studies the feature extraction process, feature meaning, and feature location of the typical CNN model used by the detector in the recognition process, forming a complementary method for patch generation. In this paper, an attention mechanism is introduced to establish the class activation graph visualization mechanism, which can visualize the key feature fields of the identified image and guide the generation and placement of adversarial patches. Figure 5 shows the process of generating a feature heatmap based on the attention mechanism. A five-layer convolution neural network is used to extract the features of the objects in the image, and a class score is used to establish class activation mapping to calculate the feature maps corresponding to different detectors.

In this paper, we use the idea of a class activation map to building a heatmap based on the attention mechanism. This scheme uses the idea of NETWORK IN NETWORK and uses global average pooling to replace the full connection layer in CNN. GAP is a special average pool layer. The reasons for adopting GAP are as follows.

Because there is no full connection layer, there is no need to adjust the size of high-definition and high-resolution images, resulting in the loss of image details. Any size image input makes it possible to support applications such as high-resolution satellite images in the future. Second, the introduction of GAP makes full use of spatial information. When the algorithm is replaced by the full connection layer, the excessive parameters of the full connection layer are deleted, which improves the robustness of the algorithm and avoids the overfitting phenomenon. Finally, in the MLPConv structure, the number of feature graphs is the same as the number of categories. After the operation of global average pooling, the results calculated by the softmax layer have a clear corresponding relationship between the feature map and the feature vector. That is, categories confidence maps are generated.

Finally, ReLU is used to calculate the output of weighted sum to ensure that the output of the algorithm only focuses on the pixel features related to classification. From the visualization of heatmap, we can see the key characteristic field of the detection target intuitively, which provides the evidence for the location, angle, and size of the patch.

3.3. Construct Loss Function. The purpose of this paper is to construct a set of adversarial patch generation algorithms which can be used to cheat the target detector. A large number of studies have proved the feasibility of adversarial patching. For objects with little change in shape, we can cheat the object detector by generating adversarial examples, but for objects with variable shapes, it is not easy to produce adversarial examples. We choose to cheat the target detector by patch, which is important because it does not need to modify the detected target directly.

At the same time, an attacker can produce different adversarial patches according to different targets in the

(a) (b) (c)

Figure 3: (a) Adversarial patch from our paper. (b) A toaster patch for inception-V3. (c) Adversarial patch from fooling, which looks like a bear.

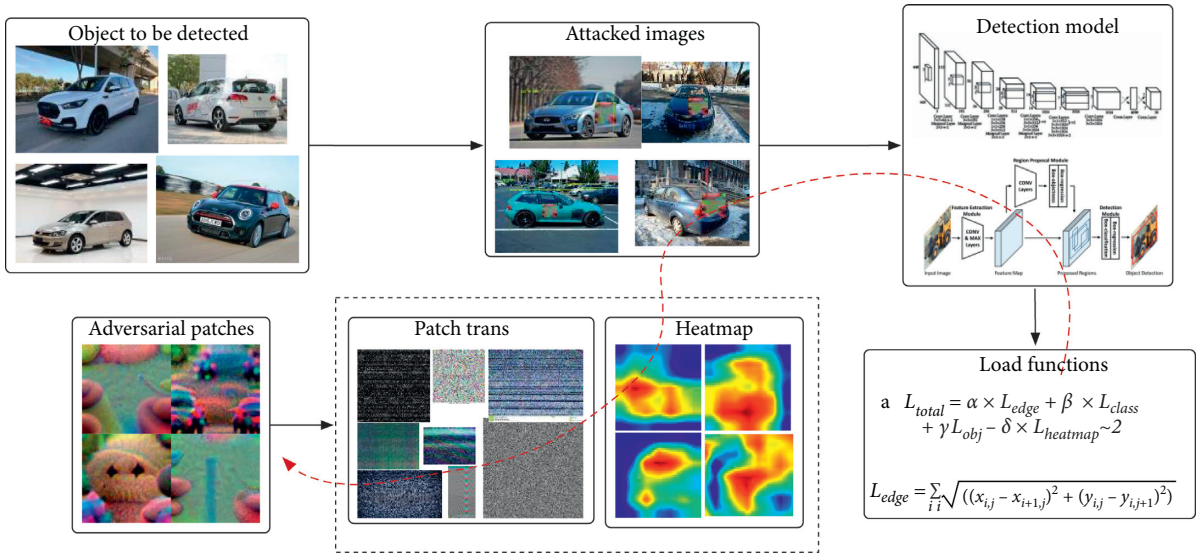


Figure 4: The overview of the pipeline of our algorithm. The black arrow represents the basic process of generating an adversarial patch. The red dotted line is used to iteratively calculate the patch display form and location through backpropagation according to the loss function.

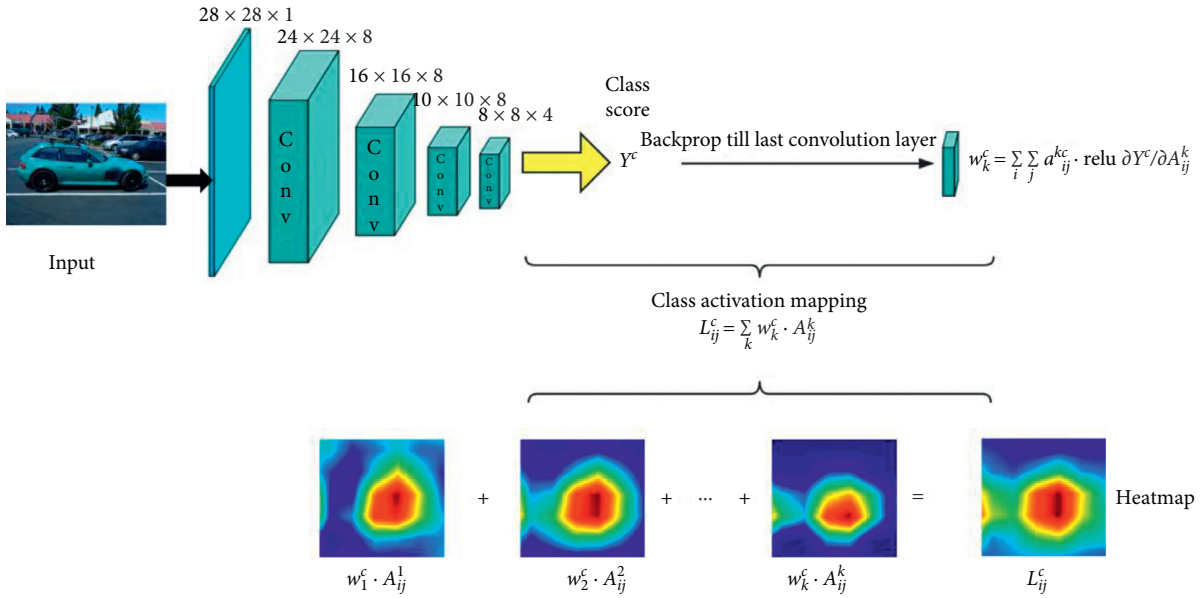


Figure 5: In this model, images are transformed into tensor form as the input of the visualization network. The final heatmap is obtained by feature extraction of each layer and weight. By modifying the last layer of network structure, the requirement of input image size is reduced.

process of the attack, which has a high degree of confusion. From a practical point of view, digital adversarial patches are more widely used than physical ones. They can be widely spread through the network and affect the judgment of search engines and object detectors. The loss function is optimized by iteratively updating the gradient. In the process of generating an adversarial patch, this paper carries out random translation, deformation, scaling, and other operations on the patch covered on the image to further improve the robustness of the patch and calculate its gradient loss.

Based on the principal analysis of the detector, the purpose of this paper is to greatly reduce the confidence of the detector in identifying the car. For the detector, when the confidence level of the detection is low enough, the target will weaken into a part of the background; through the optimization of classification score and object score, the attack effect is adjusted. The optimization process needs to consider the following factors:

L_{edge}: the transition between patch edge and image should be smooth as possible. That is, the difference between patch and image needs to be optimized.

$$L_{\text{edge}} = \sum_{i,j} \sqrt{\left(\left(x_{i,j} - x_{i+1,j}\right)^2 + \left(y_{i,j} - y_{i,j+1}\right)^2\right)}. \quad (4)$$

$L_{\text{heatmap}\sim 2}$: based on the key features of heatmap and the minimum value of L_2 norm of the patch, the randomly generated patch takes the information in heatmap as initialization information.

L_{obj}: the goal of this algorithm is to make the target detector unable to recognize the vehicle in the image by constructing an adversarial patch; in this step, the L_{obj} variable needs to be minimized to reduce the score that the object detector considering that there are no objects in the detection field;

L_{class}: optimize the classification score to reduce the score of the objects detected by the detector as vehicles and further control the classification score in the process of patch generation.

Based on the definition above, the total loss function is as follows:

$$L_{\text{total}} = \alpha \times L_{\text{edge}} + \beta \times L_{\text{class}} + \gamma L_{\text{obj}} - \delta \times L_{\text{heatmap}\sim 2}. \quad (5)$$

The total loss function takes the sum of the values of the three loss functions, where α, β, γ, and δ are the equation parameter, which can be set according to the expert opinion in general. By optimizing and calculating the total loss function, the optimal value of the loss function is obtained.

3.4. Data Training Strategy. In the real environment, the angle and size of the car are not fixed in the photos taken by cameras, mobile phones, and even high-resolution satellites. In the process of recognition, recognition models such as YOLOv2 and Faster-RCNN often intercept the identified target first and then establish the bounding box. Vehicle images input into the classifiers may overlap with other vehicles due to different vehicle types. Therefore, attackers must enhance their robustness through iterative optimization and multiple transformation combinations. In recent years, with the increasing research of adversarial attacks, there are more and more methods of data training and patch application in physical objects.

3.5. Experiments. We verify the effectiveness of the proposed algorithm framework. The algorithm is trained and verified based on the coco2017 data set. Coco2017 is a large image dataset released by Microsoft, which is designed for object detection, segmentation, human key point detection, semantic segmentation, and caption generation. The database collects data through extensive use of Amazon Mechanical Turk. Coco datasets now have three annotation types: object instances, object keypoints, and image captions. The coco2017 dataset includes three subsets: train (118287 images), Val (5000 images), and test (40670 images), with a total of 80 classes. Ground truth is provided for train and Val sets, but ground truth is not provided for the test set.

In our experiment, label files are used to mark the precise coordinates of each segmentation + bounding box, and the precision is two digits after the decimal point. The label of a target is as follows:

{“segmentation”: [[392.87, 275.77, 402.24, 284.2, 382.54, 342.36, 375.99, 356.43, 372.23, 357.37, 372.23, 397.7, 383.48, 419.27, 407.87, 439.91, 427.57, 389.25, 447.26, 346.11, 447.26, 328.29, 468.84, 290.77, 472.59, 266.38], [429.44, 465.23, 453.83, 473.67, 636.73, 474.61, 636.73, 392.07, 571.07, 364.88, 546.69, 363.0]], “field”: 28458.996150000003, “iscrowd”: 0, “image_id”: 503837, “bbox”: [372.23, 266.38, 264.5, 208.23], “category_id”: 4, “id”: 151109}.

3.6. Analyse Key Feature Fields and Output Heatmap. YOLOv2 can simultaneously predict multiple bounding boxes and their class probabilities with a single convolution network. In the process of recognition, firstly, the detected image will be reduced and trimmed according to the detection target, and then multiple bounding boxes will be established for detection. After verification, although YOLO can quickly identify the target in the image, it is not accurate in locating certain objects, especially small objects. But this has little effect on the detection of vehicles in this paper.

This paper leverages the method mentioned above to calculate the feature mapping of the output of the convolution layer. In essence, these feature maps have a certain spatial correspondence with the original image. Finally, the feature map of the convolution output of the last layer is processed and redrawn to the original image to get the thermal map. The highlighted field in the heatmap is the distribution field of the key features in the network model.

Figure 6 shows the heatmap generated from the dataset based on the detection model. Among them, picture 1 is from the Internet; picture 2 below is from a mobile phone shooting.

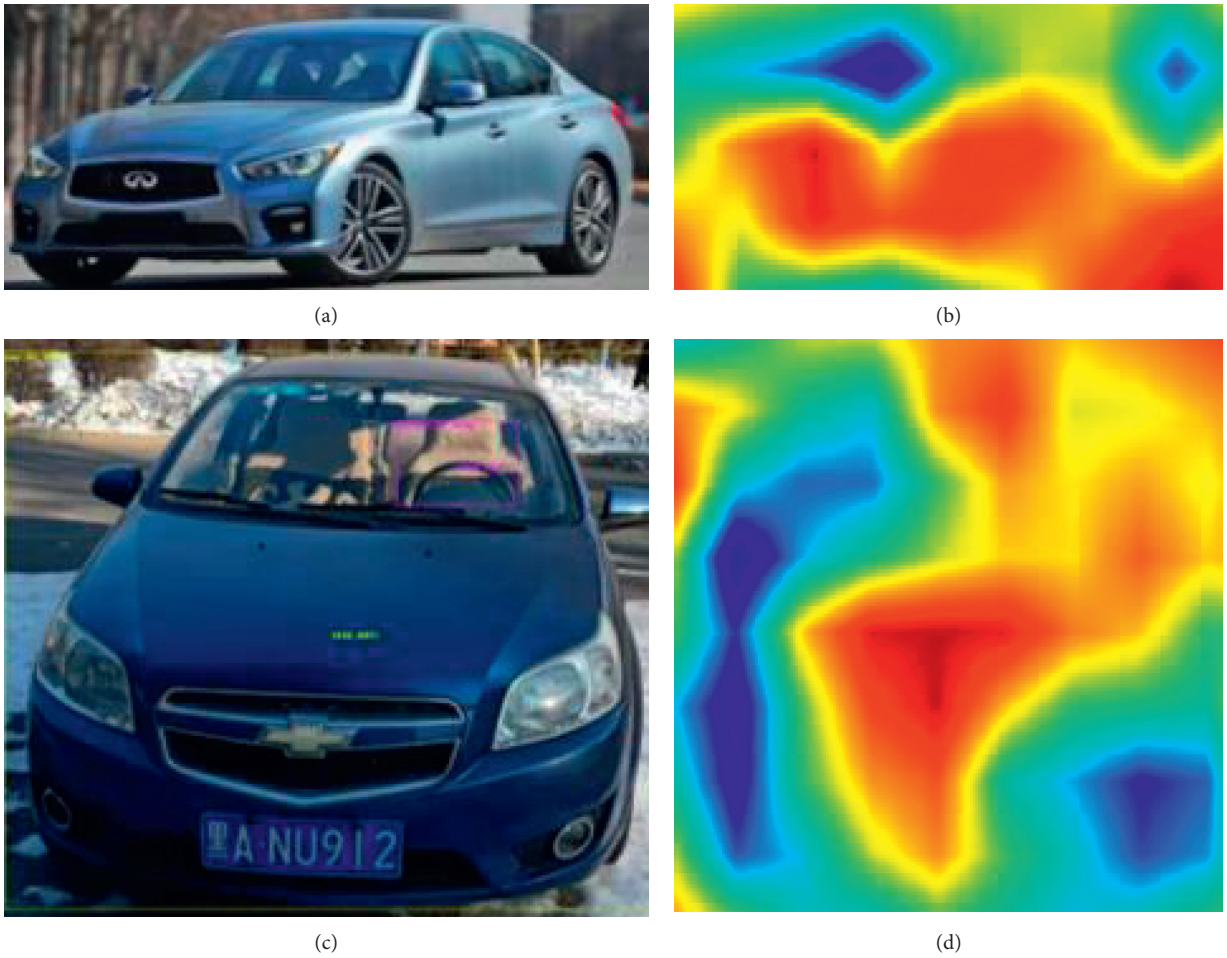


Figure 6: The two pictures above and the following two pictures, respectively, demonstrate the heatmap generated in actual operation.

3.7. Iterative Generated Adversarial Patches. To generate adversarial patches, one part of the parameters needs to be fixed, and another part of the parameters should be optimized to reduce the loss function. Because the reduction of the loss function actually affects the calculation of the object score and class score of the monitoring model, as long as the score is reduced to a certain extent, the attack effect can be achieved. If the loss function fails to converge or the target score is not well controlled in the iterative optimization process, it will lead to the misclassification of attacking missions. In this case, the attack is still successful.

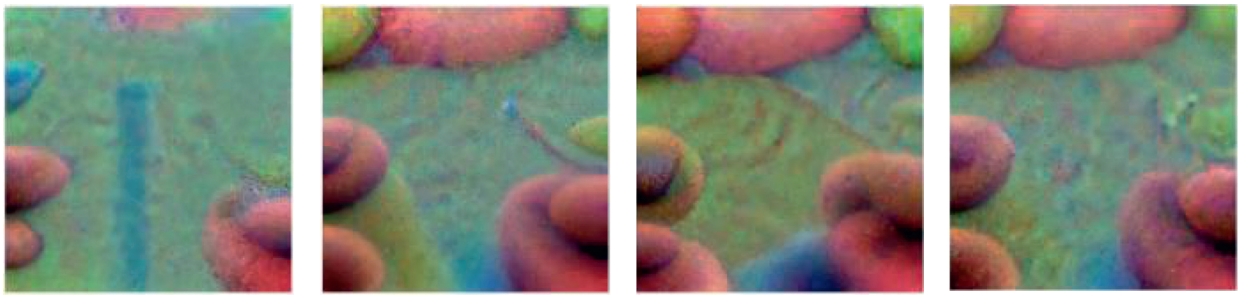
Based on the analysis of the YOLO model, the target score is composed of object score and class score. The formula is as follows:

$$\text{targetScore} = \varphi \times \text{objeScore} + \phi \times \text{classScore}. \quad (6)$$

The parameters φ and ϕ represent the weight bias of the two scores adjusted according to experience in the training process. According to the result of the formula (4), with the increased number of training batches, although the human eye cannot tell the differences of patch appearances and the patch itself does not have the real physical meaning, its attack effect is improving. Figure 7 shows the adversarial patches of four batches of training with each batch training 1000 times.

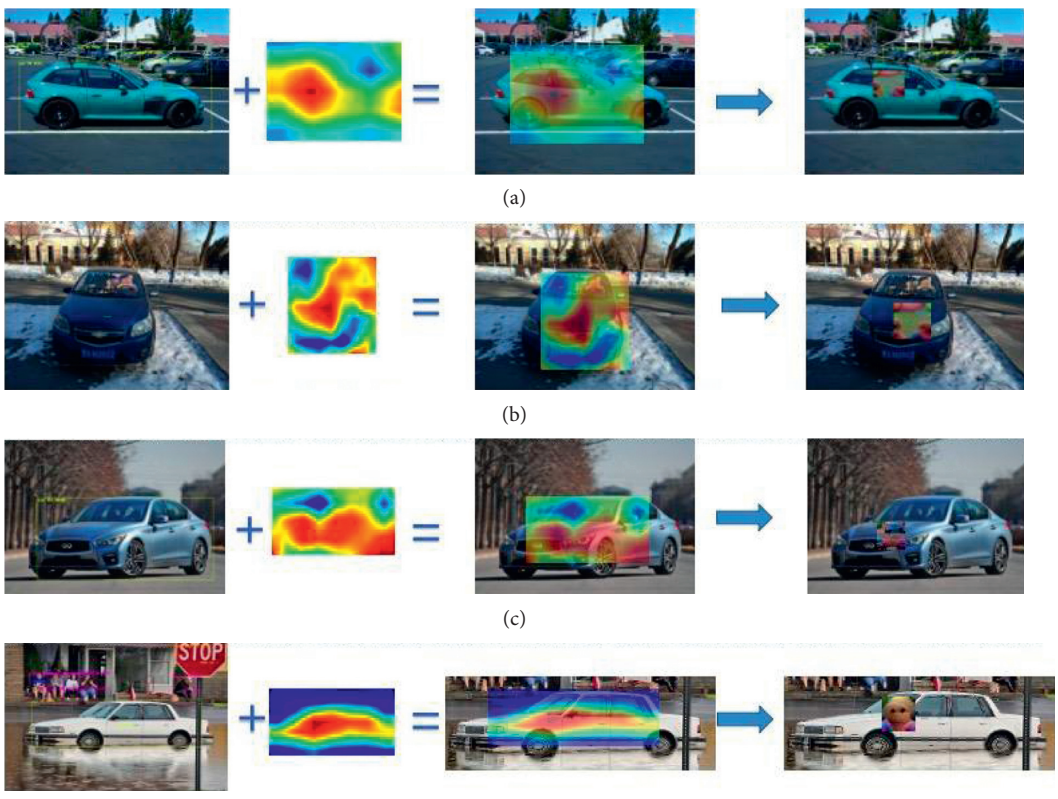
3.8. Attention-Guided Patches. In this paper, we verified the generated adversarial patches by the images from different sources. Taking the picture as input, the image is processed through the detector network to identify the vehicle in the picture. The method proposed in this paper is to optimize the location, size, and angle of the adversarial patch by establishing a heatmap based on an attention mechanism. According to the introduction above, heatmap generation algorithm itself is a kind of algorithm based on recognizer network structure. Therefore, the heatmap generated for YOLOv2 is different from that generated for the Faster-RCNN model in size and layout. Therefore, the adversarial patch generated in this paper will be different in appearance and placement due to different detectors. In essence, our algorithm is a white-box-based attack algorithm. Meanwhile, the generated patch is used in Faster-RCNN as a black-box test to verify the effectiveness of this method.

As shown in Figure 8, the images in the verification experiment are mainly from four categories: images from Microsoft's coco2017 dataset; images taken by mobile phones; images downloaded from the Internet; and news pictures, in which the car is partially covered by water. The standard YOLOv2 model can accurately identify the car in the picture. According to different background



(a) (b) (c) (d)

Figure 7: Four batches of adversarial patches generated by our work.

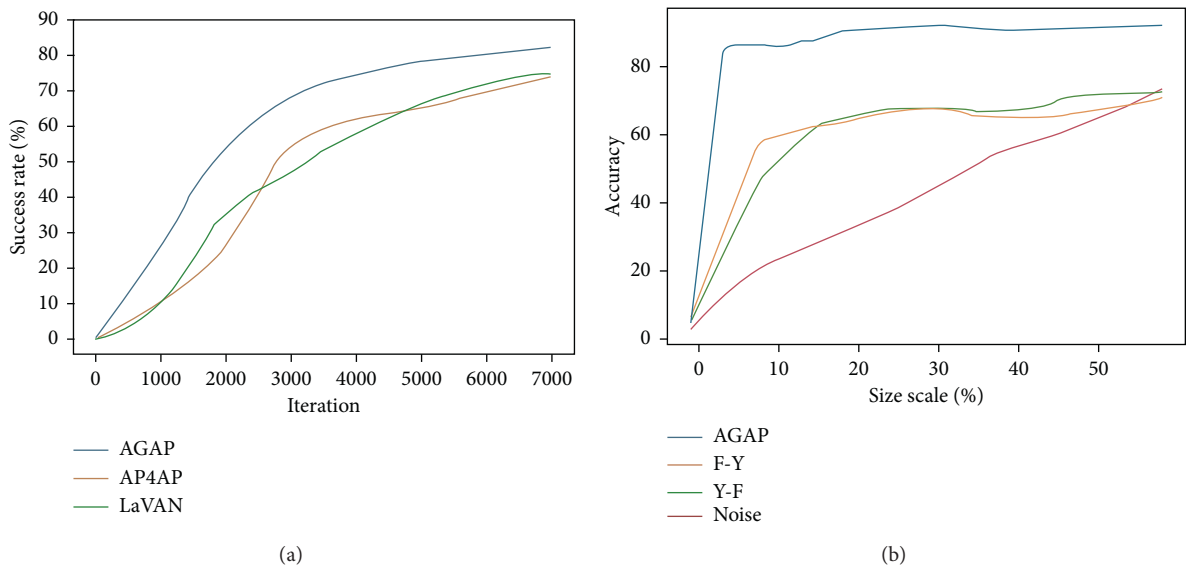


(a) (b) (c) (d)

Figure 8: The flow chart generated by the attention mechanism is guided by the attention mechanism. On the images from different sources, heatmap with key features is used to limit the generation of the adversarial patch. This method can achieve the purpose of making objects invisible to a certain detector.

environments and vehicle locations, we first analyse the feature distribution extracted from the detector network and visualize it to generate heatmap. Because the detector segments the target image before detecting, the shape of the generated heatmaps is different. Then, based on the target image, we generate two kinds of adversarial patches with different appearance features. According to the location and size of the dark-red field in the heatmap, we calculate the placement position of the patches as the constraint of the loss function. According to the experimental results, it can be used as the initial location of adversarial patch near the high heat field where the heatmap and image overlap, which can generate successful patches more efficiently.

Our experimental results show that under the guidance of the attention mechanism, the interactive patch generated in this paper has strong aggressiveness and can hide the car. We have noticed that not all environments and angles can be effective for cars. In some cases, the attack effect is not ideal because the convergence speed of the loss function slows down or the feedforward gradient disappears. Then, by adjusting the value of the class score, the effect of misclassification can be produced. At the same time, replacing



(a) (b)

Figure 9: (a) Shows the convergence speed of different patch generation algorithms under the same iteration times. In terms of hiding effect, this method is more stable and can generate available patches faster. (b) The performance of the algorithm in terms of transferability. With the increase of the adversarial patch field, the effect increases.

our generated patch with a random noise patch in the same position will not produce an adversarial effect.

3.9. Evaluations. In this paper, we propose a method of generating digital adversarial patches based on attention guidance, which can achieve the goal of hiding or misclassifying the objects in the target images. Compared with the existing research, it has obvious advantages. The existing methods of hiding targets mainly aim at scenes that are not foldable. The distance and relative position are relatively fixed, such as STOP signs. According to the existing research, the accuracy of the effective patch in the display will be greatly reduced due to the complexity of printing accuracy and illumination. Therefore, the printing of adversarial examples is also a research hotspot. Generally speaking, the main idea of the algorithm is to reduce the confidence level of the target detector. The optimization process is sensitive to the recognizer network, so it can be improved from the generation method of the patch and the structure of the detector. In terms of difficulty, because the main method is to optimize the feedback gradient, the local optimal solution may not converge, resulting in the failure to effectively hide the target.

The existing research is based on the position of the bounding box in YOLOv2 to locate the position of the patch. The initial size and position are generated randomly. The method proposed in this paper analyses the structure of the detector, extracts the key features in the process of target recognition based on the position, size, angle, and shape of these features. Attention mechanism is introduced to guide the formation of the patch. The convergence speed of the algorithm is faster when similar algorithms are used to generate the adversarial patch with the same effectiveness. The analysis is shown in Figure 9(a) below. The graph shows in 7000 training times, which shows that the algorithm proposed in this paper can achieve a stable convergence position quickly when the first 7000 times are trained, and other algorithms need more trial iterations in the process of generating patch.

About transferability testing, our paper improves the compatibility with other models by adjusting the parameters of the similar detector model. The robustness of the patch is improved by rotating and scaling the image. In this paper, the reverse patch, which is trained by the YOLO model, has been tested in the Faster-RCNN model. The transferability of patch trained in Faster-RCNN in the YOLO model is also tested, as shown in Figure 9(b). The results show that in terms of detection accuracy, although a lot of decline leads to some of the false classification results, the recognition accuracy of Faster-RCNN is still greatly reduced. The blue line in the figure is the algorithm AGAP proposed in this paper. The orange and green lines are transferability effects. Red represents the effect of noise generated patches on image hiding.

Whether the heatmap and adversarial patch based on YOLOv2 can be applied in other recognition networks also concerns us. In this paper, the same algorithm is used to calculate the heatmap of Fast-RCNN, and the effective patch of YOLOv2 is directly used to apply, but the effect is not ideal. Only about 30% of the cases are misclassified, or the image in the patch can be recognized as a target. Meanwhile, success rate of our algorithm to YOLOv2 reaches 88.7%. The comparison table of algorithm iteration and convergence is as Table 1, which includes AGAP, AP4AP, and LaVAN.

Through the research and verification of real data, we have accumulated the experience of analysing state-of-the-art detector and generating adversarial examples. It is of great significance to analyse the principle of deep neural networks in the detection model to improve the robustness and

Table 1: Comparison table of algorithm iteration and convergence.

IterTime	1000	3000	5000	7000
AGAP	27.2% (58/216)	63.7% (137/216)	80.1% (137/216)	82.9% (179/216)
AP4AP	10.4% (21/206)	45.2% (93/206)	68.2% (140/206)	81.1% (167/206)
LaVAN	15.4% (34/224)	52.1% (117/224)	76.2% (171/224)	84.5% (189/224)

antiattack of the detection model itself. In this paper, we propose an attention mechanism guided algorithm to generate an adversarial patch, which can hide and misclassify the car in an image in a controlled environment.

4. Conclusions

According to recent researches, the generation of adversarial examples is not a special game of wisdom but the inevitable result of the unique deep neural network structure. In this paper, we present an algorithm to generate adversarial ailing patches which can cheat the vehicle detector. This method takes into account the various conditions of the car in the image, including illumination, position, angle, colour, and other factors. Through the optimization of the image, the value of the loss function is continuously reduced, the confidence of the detected target is reduced, and the discrimination between the detected target and the background is reduced so as to cheat the detector.

Meanwhile, we propose a strategy to generate adversarial patches based on the attention mechanism. By extracting the key feature vectors of the car in the images, we can calculate and visualize the feature aggregation field and guide the generation and placement of adversarial patches. Finally, through a number of experiments, it is verified that in different application environments, the way of attention guidance can quickly generate adversarial patches. Experimental results show that the adversarial patches generated in this paper can attack the targeted vehicle well. After adjusting the parameters, the target can be misclassified.

The existing adversarial patch generation algorithms mainly reduce the correct classification of classifiers by randomly generating interference images. In the process of generating random adversarial patches, each pixel in the image to be detected has the same important weight, so it needs to iterate many times in the calculation process. The generated adversarial patch makes the detector unable to locate all the targets, instead of hiding an object that needs to be hidden; or when the distance angle changes, the adversarial patch might be invalid. This paper visualizes the distribution of key features through an attention mechanism. Compared with the traditional random generation algorithm, the accuracy of our algorithm is improved.

On the basis of this research, more and more researchers are paying attention to the algorithm and application of object adversarial patch generation in the reality scene. In actual production and life, it faces more and more complex problems to generate the adversarial patch for physical objects. However, once successful, the harm is also very severe. In the future, the application in this field will be able to produce a more effective, more robust, and more mobile adversarial patch for more state-of-the-art detectors, which should attract our attention.

Data Availability

All data underlying the results are available as part of the article, and no additional source data are required.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper was supported by 2020 Information Security Software Project, Network Threat Depth Analysis Software, and 2019 Industrial Internet Innovation and Development Engineering, On-Site Emergency Detection Tools in the Field of Industrial Internet Security (KY10600200021).

References

[1] Y. Wang, L. Haoran, K. Xiaohui et al., "Towards a physical-world adversarial patch for blinding object detection models," *Information Sciences*, vol. 556, 2020.

[2] C. Chen, O. Li, C. Tao, A. J. Barnett, J. Su, and C. Rudin, "This looks like that: deep learning for interpretable image recognition," *Advances in Neural Information Processing System*, vol. 1, 2019.

[3] J. Choi, D. Chun, H. Kim, and H. J. Lee, "Gaussian YOLOv3: an accurate and fast object detector using localization uncertainty for autonomous driving," in *Proceeding of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 502–511, IEEE, Seoul, Republic of Korea, October 2019.

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proceeding of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, IEEE, Venice, Italy, October 2017.

[5] M. Xue, C. Yuan, J. Wang, and W. Liu, "DPAEG: a dependency parse-based adversarial examples generation method for intelligent Q&A robots," *Security and Communication Networks*, vol. 2020, no. 2, 15 pages, Article ID 5890820, 2020.

[6] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceeding of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525, IEEE, Honolulu, HI, USA, July 2017.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[8] A. Kwasigroch, M. Grochowski, and A. Mikolajczyk, "Neural architecture search for skin lesion classification," *IEEE Access*, vol. 99, p. 1, 2020.

[9] N. Moritz, B. Kollmeier, and J. Anemuller, "Integration of optimized modulation filter sets into deep neural networks for automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2439–2452, 2016.

[10] Y. Fu, Y. Wei, G. Wang et al., "Self-similarity grouping: a simple unsupervised cross domain adaptation approach for person re-identification," 2019, http://arxiv.org/abs/1811.10144.

[11] K. Eykholt, I. Evtimov, E. Fernandes et al., "Physical adversarial examples for object detectors," 2018, http://arxiv.org/abs/1807.07769.

[12] L. Yujia, Z. Weiming, and Y. Nenghai, "Protecting privacy in shared photos via adversarial examples based stealth," *Security and Communication Networks*, vol. 2017, Article ID 1897438, 15 pages, 2017.

[13] J. Hayes, "On visible adversarial perturbations & digital watermarking," in *Proceeding of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1678–16787, IEEE, Salt Lake City, UT, USA, June 2018.

[14] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[15] D. Lang, S. Huang, Y. Cheng et al., "A state space abstract algorithm of incremental data recognition based on model checking," *Journal of Computational Information Systems*, vol. 10, no. 4, pp. 1731–1742, 2014.

[16] H. Chang, J. Lu, F. Yu et al., "PairedCycleGAN: asymmetric style transfer for applying and removing makeup," in *Proceeding of the CVF conference on computer vision and pattern recognition (CVPR)*, IEEE, Salt Lake City, UT, USA, July 2018.

[17] J. Marniemi and M. G. Parkki, "NO need to worry about adversarial examples in object detection in autonomous vehicles," 1975, http://arxiv.org/abs/1707.03501.

[18] B. Yda, C. Xz, A. Jz et al., "Mask-guided noise restriction adversarial attacks for image classification-ScienceDirect," *Computers & Security*, vol. 560, p. 100, 2020.

[19] L. Xu, X. Wang, W. Liu, and B. Feng, "Cascaded boundary network for high-quality temporal action proposal generation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 10, pp. 3702–3713, 2020.

[20] P. Isola, J. J. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image translation with conditional adversarial networks," in *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976, IEEE, Honolulu, HI, USA, July 2017.

[21] S. Kim, J. Jang, and O. K. Chang, "A run-to-run controller for a chemical mechanical planarization process using least squares generative adversarial networks," *Journal of Intelligent Manufacturing*, vol. 2020, pp. 1–14, 2020.

[22] L. Jiang, K. Qiao, Q. R. uoxi et al., "Cycle-consistent adversarial GAN," 2020, http://arxiv.org/abs/1904.06026.

[23] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: generalized gradient-based visual explanations for deep convolutional networks," in *Proceeding of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 839–847, IEEE, Lake Tahoe, NV, USA, March 2018.

[24] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, 2020.

[25] Đković, O. Golubitsky, and I. S. Kotsireas, "Some new orders of Hadamard and skew-Hadamard matrices," *Journal of Combinatorial Designs*, vol. 22, no. 6, pp. 270–277, 2014.

[26] K. Eshghi and M. Kafai, "The CRO kernel: using concomitant rank order hashes for sparse high dimensional randomized feature maps," in *Proceeding of the IEEE International Conference on Data Engineering*, IEEE, Helsinki, Finland, May 2016.

[27] F. Chen, N. Wang, J. Tang, and F. Zhu, "A feature disentangling approach for person re-identification via self-supervised data augmentation," *Applied Soft Computing*, vol. 100, no. 1, Article ID 106939, 2021.

[28] A. Liu, X. Liu, J. Fan et al., "Perceptual-sensitive GAN for generating adversarial patches," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1028–1035, 2019.

[29] P. Wang, B. Jiao, Y. Lu et al., "Vehicle re-identification in aerial imagery: dataset and approach," in *Proceeding of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 460–469, IEEE, Seoul, Republic of Korea, September 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6659022, 13 pages
https://doi.org/10.1155/2021/6659022

Research Article

Deep-Feature-Based Autoencoder Network for Few-Shot Malicious Traffic Detection

Mingshu He,[1] **Xiaojuan Wang**,[1] **Junhua Zhou**,[2] **Yuanyuan Xi**,[3] **Lei Jin**,[1] **and Xinlei Wang**[1]

[1]*School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*State Key Laboratory of Intelligent Manufacturing System Technology, Beijing Institute of Electronic System Engineering, Beijing 100854, China*
[3]*School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm SE-10044, Sweden*

Correspondence should be addressed to Xiaojuan Wang; wj2718@163.com

Received 27 December 2020; Revised 1 February 2021; Accepted 15 March 2021; Published 27 March 2021

Academic Editor: Liguo Zhang

With the increase of Internet visits and connections, it is becoming essential and arduous to protect the networks and different devices of the Internet of Things (IoT) from malicious attacks. The intrusion detection systems (IDSs) based on supervised machine learning (ML) methods require a large number of labeled samples. However, the number of abnormal behaviors is far less than that of normal behaviors, let alone that the shots of malicious behavior samples which can be intercepted as training dataset are actually limited. Consequently, it is a key research topic to conduct the anomaly detection for the small number of abnormal behavior samples. This paper proposes an anomaly detection model with a few abnormal samples to solve the problem in few-shot detection based on convolutional neural networks (CNN) and autoencoder (AE). This model mainly consists of the CNN-based supervised pretraining module and the AE-based data reconstruction module. Only a few abnormal samples are utilized to the pretrain module to build the structure of extracting deep features. The data reconstruction module simply chooses the deep features of normal samples as training data. There also exist some effective attention mechanisms in the pretraining module. Through the pretraining of small samples, the accuracy of abnormal detection is improved compared with merely training normal samples with AE. The simulation results prove that this solution can solve the above problems occurring in network behavior anomaly detection. In comparison to the original AE model and other clustering methods, the proposed model advances the detection results in a visible way.

1. Introduction

Network application plays an indispensable role in people's lives with a large number of devices connected to the Internet; network security is attracting greater attention. With the continuous development of network technology, cyber threats are increasingly complex. According to the statistical prediction, the devices of the Internet of Things will increase from 27 billion to 75 billion by 2025, constituting a huge scale of network access devices [1]. Meanwhile, the number of network attacks will rise rapidly as well, which explains why network attack detection is widely concerned by researchers. There are some intrusion detection systems especially for identifying malicious behaviors on the Internet and raising alarms [2]. However, the detection process tends to be complicated in the real production network environment. For the time being, most methods, such as ML-based models, always require a large number of labeled samples to accomplish the training process. Nevertheless, only a limited number of malicious behaviors can be intercepted to be the training data. Hence, few-shot detection is urgently needed. More seriously, a new attack pattern can hardly be predicted previously before it appears, which requires good sensitive detection ability of the model for unknown types of malicious

network traffic. Focusing on this problem, this paper also adopts a large number of samples of normal behavior and a few abnormal behavior samples in pretraining, which is applicable to the actual few-shot learning scenario.

There are two major methods for malicious network behavior detection, traditional business approaches and ML-based ones [3]. Generally speaking, both of them need to extract the necessary information or features based on professional awareness. Deep Packet Inspection- (DPI-) based, port-based, behavior-based, and statistic-based methods are several frequently used traditional approaches [4, 5]. These methods rely on strong single service characteristics, being excellent at identifying specific threats accurately. However, if some new threats occurred, the new business features and rules need to be extracted as well, making it difficult for traditional methods to identify malicious behaviors with unstable rules and features. By contrast, ML methods usually take out business features from more various dimensions, which therefore provides better generalization.

However, from a business perspective, there is a limitation that the performance of an ML model always depends on the feature engineering results based on manual analysis, which makes our feature mining task heavy. At the same time, the training data also rely on specialized knowledge concerning cyber security and attacks. As a result, researchers need to be more concentrated on feature engineering work, which means a great challenge to the efficiency of anomaly detection. Besides, some methods try to encrypt in the process of information transmission [6, 7], which makes the feature extraction of encrypted traffic more difficult. Compared with the traditional machine learning algorithm, the deep learning (DL) model has its own advantages in extracting features. For example, the CNN-based model can directly obtain the deep features without manual feature engineering, the data processed by which displays a better spatial distribution. This is one of the reasons why CNN is selected for the pretraining process. In common with all deep learning algorithms, the model proposed in this paper also eliminates the manual process of feature extraction. Except for this, the model will no longer require a mass of abnormal samples to carry out supervised learning. During the detection process, the pretraining model only needs a small number of abnormal samples to achieve a good performance of the subsequent unsupervised anomaly detection. By doing so, the problems of few-shot detection can be worked out as well.

The contributions of this paper can be summarized as follows:

(1) Proposing a network malicious behavior detection model based on CNN and AE (DFAE), which can not only achieve a good detection result in the case of a small number of abnormal training samples but also provide a certain guiding significance for the real abnormal detection scenario in which the number of normal samples is much larger than that of abnormal samples.

(2) Improving the AE detection results by a large margin by using the output of deep features from the CNN-based pretraining model as the training data.

(3) Enhancing the anomaly detection results on precision, recall, and F1-score in three available datasets by adding an effective attention mechanism into each block of CNN.

The remainder of this paper is arranged as follows. Some related works are arranged in Section 2. In Section 3, we introduce the detailed structure of the models and the specific process of the method. There is also some basic knowledge regarding the proposed model. Section 4 analyzes the results produced by the proposed model and some comparative experiments. Section 5 summarizes the paper on the whole.

2. Related Work

Network traffic as an essential means to record the network behavior process contains almost all the information of a complete access from the source host to a destination host [8] and can be captured and collected in a packet capture (PCAP) file [9] in most implementations. This is because of its data retention and general integrity [10, 11]. In general, extracting efficacious information and valid business features from source data is the first step in network abnormal behavior detection [12]. Similarly, this paper picks out PCAP files of three different datasets and extracts effective information from files for the following anomaly detection.

At present, there are a number of network traffic anomaly detection methods based on machine learning. Many of them pay attention to business features and adopt the approach of traditional machine learning. Meanwhile, some researchers are inclined to use neural networks to extract deep features for detection without any artificial process [5]. Also, there are studies focusing on distinct anomaly detection results output from different traditional algorithms on dataset NSL-KDD [13, 14]. Negandhi et al. employed the supervised learning algorithm to train a network attacks detection model based on random forests. The model reduces the number of input features under an effective feature selection mechanism, not only improving the running speed but also realizing a high accuracy [15]. Sarker et al. proposed an ML-based multilayered framework for the purpose of promoting the security of network system, which aims at the applicability towards data-driven intelligent decisions and protects network systems and devices from network attacks [16]. Atli proposed a traffic classification method that identifies the normal traffic and encrypted traffic by analyzing network flow based on decision tree (DT) and *K*-Nearest Neighbor (KNN) algorithms [17]. The work by D'hooge and Kayes concluded that the results of anomaly detection with machine learning algorithms as a basis are not ideal among different datasets [18], which provides a research impetus to increase the generalization of the model with limited data. Without exception, these methods cost a lot of time in feature extraction. That is why the methods rooted in the neural networks are extensively utilized in anomaly detection tasks.

The research by Zeng et al. proposed a light-weight framework without manual intervention and private

information but with the aid of deep learning for encrypted traffic classification and intrusion detection [19]. The model in [20] studied by Thapa et al. was based on classification and regression tree (CART) and CNN and performed well in 10-fold cross-validation and independent testing on dataset CIC-IDS2017 [21]. Compared with some other methods, the model brought forward by Javaid employs a self-taught learning technique on NSL-KDD and, as a result, improves the precision and recall rates [22]. Chen et al. proposed a network traffic classification model on the basis of metric learning, which gained a better performance on some available datasets by adding additive angular margin loss embedded on CNN [23]. The traffic classification method based on text convolution neural network was projected by Song et al., in which the model performs better because of adding a new loss function [24]. Zhu et al. proposed a long short-term memory- (LSTM-) based anomaly detection method [25]. Du et al. put forward a packet-based malicious payload detection and identification algorithm based on object detection deep learning network [26]. At the same time, reinforcement learning (RL) [27, 28] which is widely used in resource scheduling is applied to anomaly detection [29]. These methods based on deep learning make a lot of contributions to the network security anomaly detection tasks. However, none of them raised a good solution to the problems mentioned in Section 1, including the lack of abnormal sample markers as well as the recognition of new abnormal behaviors.

In the fields of network anomaly detection and other similar ones, some methods try to solve the class imbalance by modifying the model structure formed by ML algorithms. The problem of too few labeled samples of abnormal behaviors reveals the imbalance between negative samples and positive ones. Buda et al. analyzed the imbalance problem in image classification tasks and investigated some solutions [30]. As for Rodda and Erothi, they exerted the traditional ML methods to evaluate the effect of class imbalance on the benchmark NSL_KDD dataset [31]. Gu et al. presented a semisupervised weighted *K*-means detection method to tackle the problem that supervised learning methods need abundant labeled data [32]. On the foundation of variational autoencoder (VAE), Xu et al. concluded an unsupervised anomaly detection algorithm named Donut, which remarkably advanced the results of anomaly detection [33]. Also, some data augmentation methods are put to use to solve this problem in detection applications [34, 35]. In this paper, under the theory of data augmentation and the idea of unsupervised learning, we propose a network traffic anomaly detection model with supervised pretraining, which uses a few abnormal samples.

3. Model and Methods

3.1. Anomaly Detection Framework. This model targets attaching a superior anomaly detection result on the network traffic data in the circumstance of a few malicious behavior negative samples. In order to complete the experimental demonstration and explanation, we divide the whole procedure into three parts. As shown in Figure 1, there are data processing, supervised pretraining, and autoencoder training process. Besides, there are some comparative experiments as well. The similarity-based and *K*-means-based methods are explained later. In the figure here, one of the comparative methods, AE method mentioned in the experiment, is the third part training process whose input is the raw data without pretraining. Another one is the whole deep-feature-based autoencoder without any attention mechanisms. We also take advantage of the similarity of raw data to measure the distance between the test sample and benign sample center to identify whether the tested one is abnormal. What is more, *K*-means clustering model is used as the comparative method in the same way. The effectiveness of the proposed model is proved by the comparison of its results with the results of the four methods.

The network behavior is usually recorded as PCAP files by the means of network traffic. PCAP files consist of a series of hexadecimal numbers. Generally, it does not come easy to understand original numbers in a short time. The numbers in rigid positioning always represent particular meanings, the business significance of which thus needs to be perceived by some analysis software. Data processing is designed to convert raw data from PCAP into matrixes. The numbers with a certain length will be taken out from PCAPs to shape the data matrixes. After this step, the supervised pretraining process will train a classifier to seek for the feature space with a larger distance among samples from different categories. Then, deep features output from the pretraining module will be fed into autoencoder for data reconstruction with benign data only. When the data reconstruction model converged, samples could be imported into the model to evaluate the reconstruction effect and detect the abnormal samples. As can be seen from Figure 1, the outputs of abnormal samples (the red sample outputs from AE in the figure) are usually more different from inputs than those normal ones (the blue sample outputs from AE in the figure). Because only normal traffic data is used for training in this process, the model learns nothing but the reconstruction ability of normal traffic, which means the reconstruction process for abnormal data cannot be completed. In the autoencoder training process, the reconstruction rate threshold will be set, such as 90%, which is from 0 to 100%. If the sample reconstruction rate is not smaller than the threshold value, the sample will be regarded as a normal one; otherwise, it will be regarded as an abnormal one. These three processes mentioned above are followed by the realization of detection results. Also worth noting is the fact that just a few abnormal samples in the second pretraining process actualize the goal of achieving the anomaly detection with a small number of negative samples.

3.2. Model Structure of Deep-Feature-Based Autoencoder

3.2.1. Introduction of CNN and AE. This section will introduce the basic structure of CNN, AE, and the basic net of the proposed model. CNN has outstanding performance in many applications, especially in the image area [36, 37].

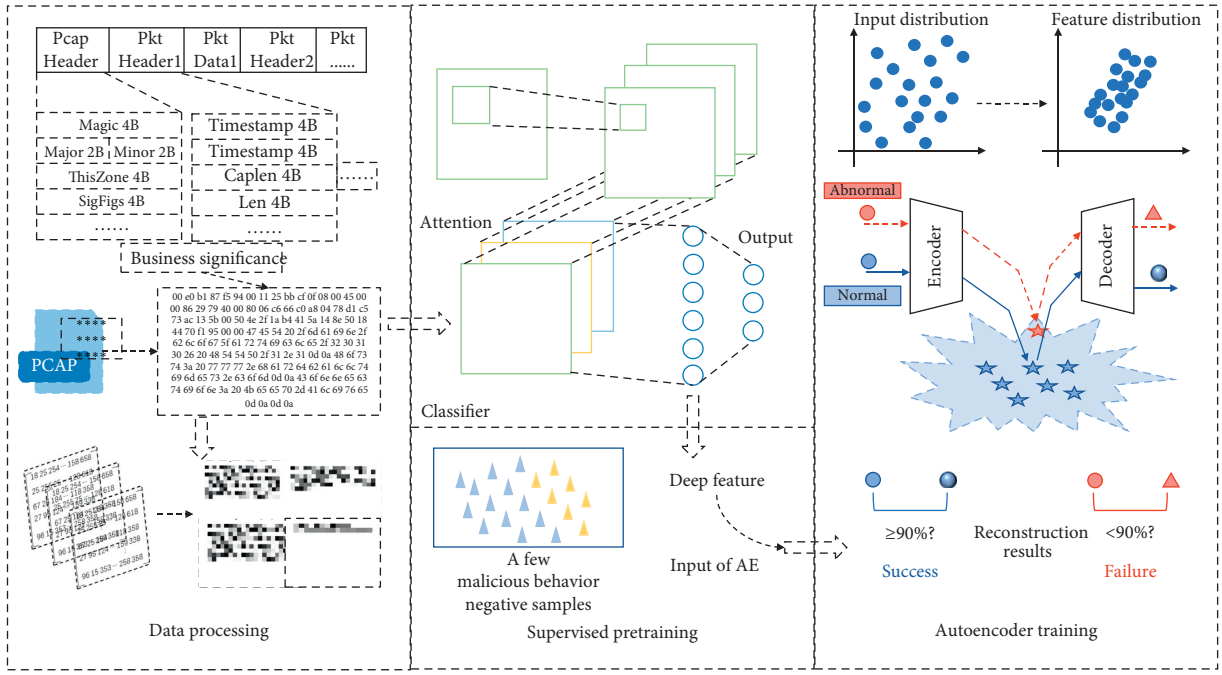


FIGURE 1: Framework of anomaly detection process. Data processing converts raw data with hexadecimal numbers to matrixes. Supervised pretraining accomplishes few-shot training and provides deep features that can be used in autoencoder training process and the data reconstruction process can be completed.

Autoencoder is an unsupervised model and is widely applied in data dimensionality reduction and feature extraction as well as data augmentation [38]. Figures 2 and 3 illustrate the simple structures of CNN and AE. A typical CNN network always contains several convolutional layers, pooling layers, and fully connected layers. Roughly speaking, the main function of a convolutional layer is to extract features, and it can obtain many effective ones without any manual intervention. Convolutional layers always serve as the beginning structure of the whole network. The convolution kernel size can be defined to extract local features with different sizes and different convolution kernels can represent different features. Then the results output from the convolutional layer are mapped by a nonlinear operation, which is usually achieved through the activation function such as rectified linear unit (ReLU), Tanh, and Sigmoid function [39, 40]. The main purpose of the pooling layer is to reduce the dimension of current data, which is actually the process of sampling. Pooling layer not only retains the main features but also greatly reduces the computation of the model. Max pooling and average pooling are two major operations in this layer. One calculates the maximum of local units and the other figures out the average in the feature map. FC layer decides which category is the true result.

An AE is made up of two layers, namely, an encoder and a decoder. AE is a genre of unsupervised learning method for dimension reduction and feature extraction. The encoder mainly encodes original data and outputs samples of dimension reduction, while the decoder mainly decodes the encoded vector to restore the original sample. Such two processes can be regarded as a data reconstruction process. CNN, FC, LSTM, and so forth are feasible to be chosen as the basic network of AE.

3.2.2. Attention Mechanism. In this paper, the model that combines CNN with AE has already reached a good result in most testings. However, it does not show outstanding performance in a tiny fraction of datasets, and there is still room for improvement Therefore, effective attention mechanisms are added to the model. When the attention mechanism is proposed in the DL model, it draws lessons from the human attention mechanism. After observing the whole situation, people tend to pay attention to some more important areas, which is just the skill the model needs to master and improve in a further step under the attention mechanism. In this paper, channel attention is attached to the end of the convolutional block [41]. We try to use three global pooling methods at the beginning of the net, namely, max pooling, average pooling, and max pooling with average pooling. Figure 4 unfolds the operation process of the attention mechanism.

According to the details, the input of a convolutional block can be defined as $\mathbf{X} \in \mathbf{R}^{H' \times W' \times C'}$ and the output is $\mathbf{M} \in \mathbf{R}^{H \times W \times C}$, where W', W, H', and H are the widths and heights of the input and output results of the convolutional blocks. C' and C denote the input and output channel sets of a block. We use $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_c]$ to refer to the learned set of filter kernels, where $\mathbf{v}_c$ is the parameter of the *c-th* filter. Hence, the output can be written as $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, \ldots \mathbf{m}_c]$, where

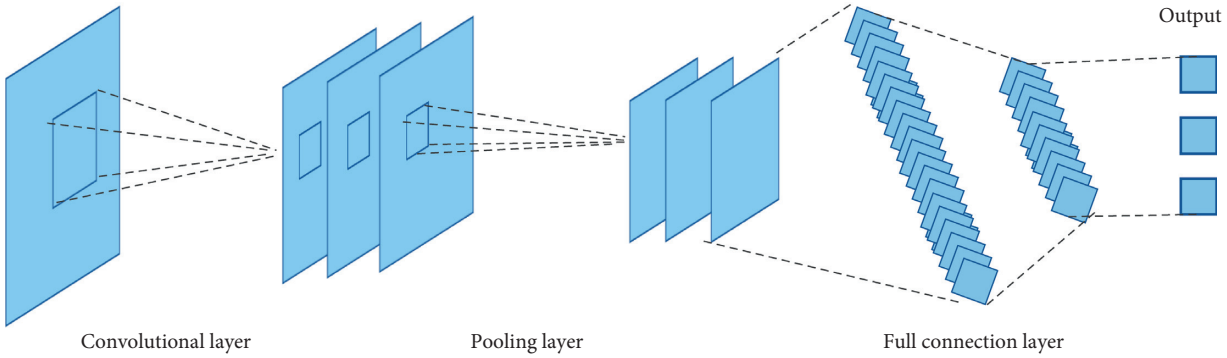


Figure 2: CNN basic model.

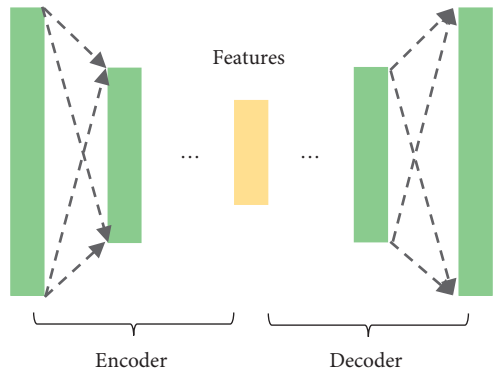


Figure 3: AE basic model.

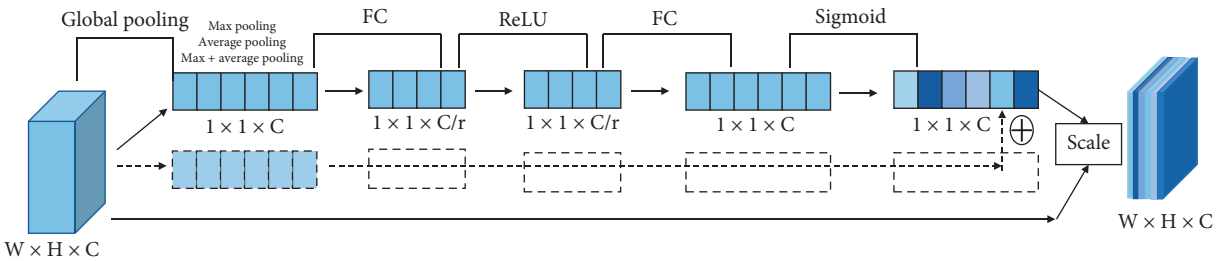


Figure 4: Structure of channel attention in the proposed model. Attention mechanism with average pooling adopted in the paper is indicated with solid rectangles and other attention mechanisms with max pooling or max and average pooling not used are indicated with dotted rectangles.

$$\mathbf{m}_c = \mathbf{v}_c * \mathbf{X} = \sum_{s=1}^{C'} \mathbf{v}_c^s * \mathbf{x}^s. \tag{1}$$

The symbol $*$ represents the convolution; $\mathbf{v}_c = [v_c^1, v_c^2, v_c^3, \ldots, v_c^{C'}]$. Next, the output of the convolutional block $\mathbf{M}$ is fed into the attention module. Firstly, the pooling operation converts $\mathbf{M}$ to size $1 \times 1 \times C$ as $\mathbf{Z}_1 = [z_1^1, z_1^2, z_1^3, \ldots, z_1^C]$, where z_1^C is calculated by

$$z_1^C = \frac{1}{H * W} \sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{m}_c(i, j), \tag{2}$$

$$z_1^{C'} = \mathrm{Max}\left(\mathbf{m}_{i,j}\right). \tag{3}$$

Equations (2) and (3) denote two different global pooling methods: average pooling and max pooling. Figure 4 shows the evaluation of the effects under the three different situations of using merely max pooling, using merely average pooling, and using average pooling with max pooling together in experiments. The average pooling is turned out to be the best one. Therefore, the other two methods will not be discussed in detail here. After that, the data will cross two fully connected layers with ReLU and Sigmoid. We define them as $\mathbf{Z}_2 = [z_2^1, z_2^2, z_2^3, \ldots, z_2^{C/r}]$ and $\mathbf{Z}_3 = [z_3^1, z_3^2, z_3^3, \ldots, z_3^C]$, where r is the parameter and it is defined as 16 here. If $\mathbf{F}_{\mathrm{fc}}$ is the fully connected operation, $\mathbf{F}_{\mathrm{relu}}$ denotes ReLU activation process and $\mathbf{F}_{\mathrm{sigmoid}}$ denotes Sigmoid activation process; $\mathbf{Z}_2$ and $\mathbf{Z}_3$ can be calculated as

$$\mathbf{Z}_2 = \mathbf{F}_{\text{relu}}\left(\mathbf{F}_{\text{fc}}\left(\mathbf{Z}_1\right)\right), \tag{4}$$

$$\mathbf{Z}_3 = \mathbf{F}_{\text{sigmoid}}\left(\mathbf{F}_{\text{fc}}\left(\mathbf{Z}_2\right)\right). \tag{5}$$

Finally, we define $\mathbf{F}_{\text{scale}}$ as the channel-wise multiplication between the scalar z_3^C and the input feature map $\mathbf{m}_c$. Moreover, the output of the attention module is $\widetilde{\mathbf{M}} \in \mathbf{R}^{H\times W\times C}$, which can be calculated as

$$\widetilde{\mathbf{M}} = \mathbf{F}_{\text{scale}}\left(\mathbf{Z}_3\right). \tag{6}$$

Here, we employ the mean squared error (MSE) L_{mse} as the loss function, which can be calculated as

$$L_{\text{mse}}(y) = \sum_i \left(y_i' - y_i\right)^2, \tag{7}$$

where i is the prediction probability of i-th category, y_i signifies prediction result, and y_i' indicates the true result. Through attention mechanism, a feature map with channel weight $\mathbf{Y}$ can be acquired, which can be regarded as a self-attention function on channels whose relationships are not confined to the local receptive field to which the convolutional filters are responsive.

3.2.3. Deep-Feature-Based Autoencoder. Sections 3.2.1 and 3.2.2 depict the basic structure of the proposed model in detail. We can divide the model into four processing modules. They are supervised deep feature extraction module (SDFE), unsupervised data reconstruction module (USDR), channel attention module (CA), and detection results output module (DROut). Actually, CA is a part of SDFE and DROut is the decision part of USDR. This detailed process is described at length in Figure 5.

As Section 4.2 described, raw data will be converted into a matrix with the data ranging from 0 to 255 whose width is 28 and height is 28. Thus, the input can be written as $\mathbf{X} \in \mathbf{R}^{28\times 28}$. Next, the data will cross the first convolution layer. The output channel is set to 32 and the convolution kernel size is 9. ReLU is the activation function here. Besides, the first pooling kernel size is 2. After all these processes mentioned above, we have $\mathbf{M}_1 \in \mathbf{R}^{10\times 10\times 32}$. The calculation process of each layer can be generalized as follows:

$$\mathbf{M}_1 = \mathbf{F}_{\text{pooling}\times 2}\left(\mathbf{F}_{\text{relu}}\left(\mathbf{F}_{\text{conv}\times 32\times 9}\left(\mathbf{X}\right)\right)\right), \tag{8}$$

where $\mathbf{M}_1 \in \mathbf{R}^{10\times 10\times 32}$ denotes the results, $\mathbf{F}_{\text{conv}}$ means a convolution operation, and $\mathbf{F}_{\text{pooling}\times 2}$ refers to a pooling operation with 2-size pooling kernel, the result of which will be fed into the channel attention, and the corresponding output result is $\widetilde{\mathbf{M}_1} \in \mathbf{R}^{10\times 10\times 32}$ with the channel attention weight depicted in Section 3.2.1.

After that, there is the second convolutional layer with ReLU and the second pooling layer, where the convolution kernel size is 9, the channel size is 64, and pooling kernel size is 2. What is mentioned above can be described as follows:

$$\mathbf{M}_2 = \mathbf{F}_{\text{pooling}\times 2}\left(\mathbf{F}_{\text{relu}}\left(\mathbf{F}_{\text{conv}\times 64\times 3}\left(\widetilde{\mathbf{M}_1}\right)\right)\right), \tag{9}$$

where $\mathbf{M}_2 \in \mathbf{R}^{4\times 4\times 32}$ is the output from the second convolution block. The samples then cross the attention layer and obtain the result $\widetilde{\mathbf{M}_2} \in \mathbf{R}^{4\times 4\times 64}$ with channel attention weight. Finally, $\widetilde{\mathbf{M}_2}$ is entered into a fully connected layer to acquire the supervised classification results. When the supervised classifier training converges, a large amount of benign data will be sent to the supervised model and receive the deep features $\mathbf{Y} \in \mathbf{R}^{1\times 1024}$ reshaped from $\mathbf{M}_2$ for each sample. In SDFE, average cross entropy error (ACE) is used to be the model loss L_{ace}, which can be calculated as

$$L_{\text{ace}}(y) = \sum_i y_i' \log\left(y_i\right), \tag{10}$$

where i is the prediction probability of i-th category, y_i indicates the prediction result, and y_i' conveys the true result.

By way of this treatment, $\mathbf{Y}$ will be imported into USDR. There is an encoder and a decoder in it, as stated in Section 3.2.1. Moreover, three fully connected layers are designed in both encoder and decoder. The sizes of each output feature are 256, 64, and 20 in the encoder and 64, 256, and 1024 in the decoder where the activation functions are ReLU after the first five FC and Sigmoid after the last FC. Then the final reconstructed feature vector can be represented as $\widetilde{Y} \in \mathbf{R}^{1\times 1024}$.

In the output module, the model will evaluate the effect of the reconstruction process. During the training process, we only use benign data as the training data. When the model converges, the model will have a good reconstruction ability for benign samples, which is followed by adding test data including normal and abnormal ones into the trained model. At this time, there will be a significant difference between normal samples and abnormal ones. This paper uses Spearman's rank correlation coefficient and Pearson correlation coefficient to portray the reconstruction probability between the input $\mathbf{Y}$ and $\widetilde{\mathbf{Y}}$ of USDR. A reasonable threshold will also be set to identify whether the sample is normal or abnormal.

4. Experiments Process and Results

4.1. Data Description. In this paper, PCAP files are the raw data format from which the model gets the input value. PCAP files can be translated into a group of hexadecimal numbers. Specific hex numbers or their combination at a specific location represents a specific business significance. Generally speaking, a PCAP file consists of a certain number of traffic packets. The structure of a PCAP file and the meaning of its various positions are illustrated in Figure 6.

There is a global header in a PCAP file. As Figure 6 shows, the 4-Byte Magic content represents the beginning and tells the recognition sequence of Byte in this file. The 2-Byte Major content means the major file version number. The 2-Byte Minor content is the minor file version number. The 4-Byte ThisZone refers to the local standard time. The 4-Byte SigFigs is the accuracy of the timestamp. The 4-Byte SnapLen represents the maximum storage length. The 4-Byte LinkType content is the link type. The length of the packet header is 16 Bytes, defining the 4-Byte high-order position of capture timestamp, the 4-Byte low-order position of capture timestamp, 4-Byte data length of currently captured, and 4-Byte actual data length. Accordingly, there is a packet data

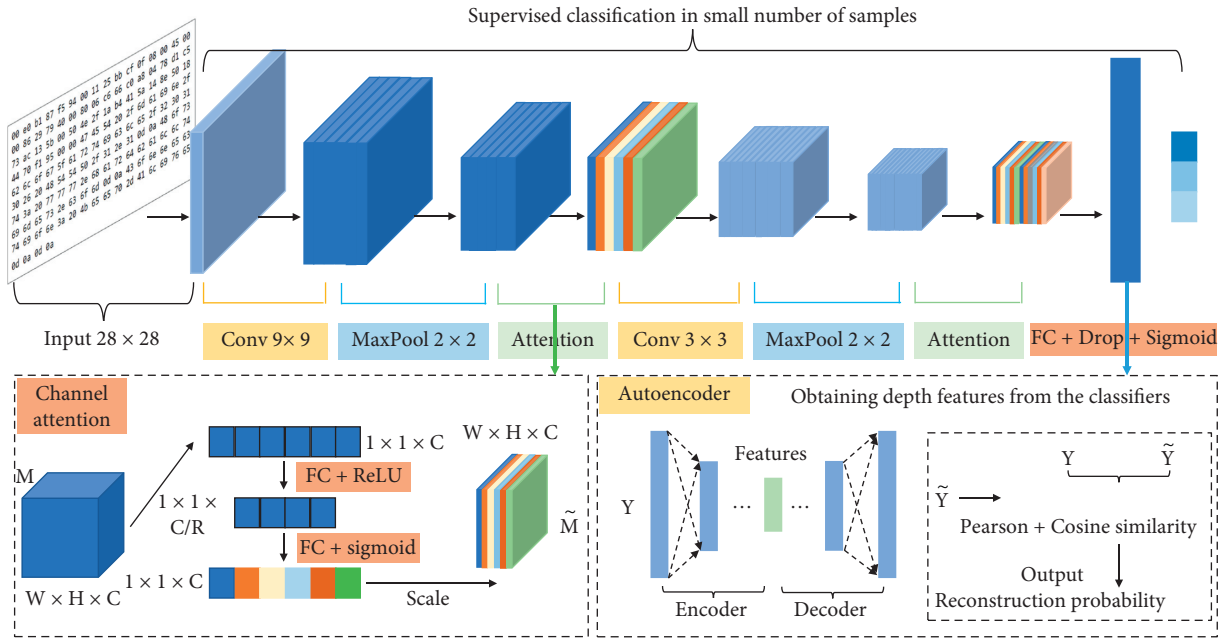


Figure 5: Model structure details of deep-feature-based autoencoder.

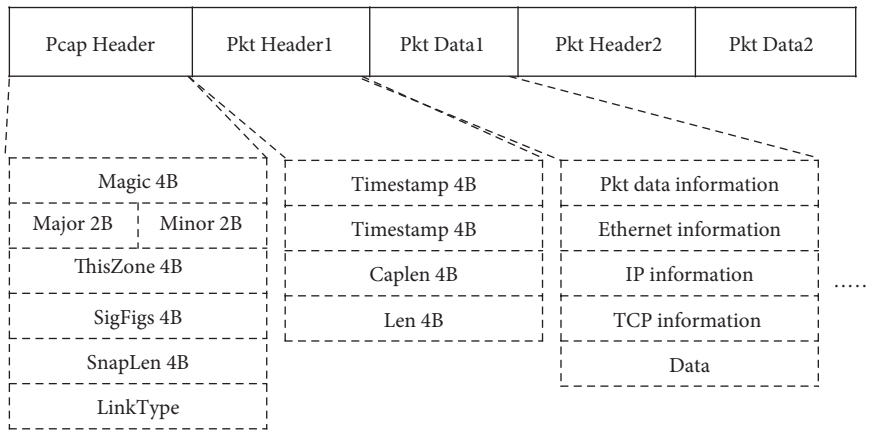


Figure 6: Structure of a PCAP file. It always consists of a file header, some packet headers, and some packet data.

area that always includes Internet Protocol (IP) information and Transmission Control Protocol (TCP) information and will be splicing of a certain number of traffic packets.

PCAP files with many traffic packets are selected as the original data source in this paper. There are three different open-source network attack datasets throughout the whole experiment, namely, CIC-IDS2017, CIC-IDS2012, and USTC-TFC2016, and they are all available through open-source addresses. In detail, CIC-IDS2017 contains over 2.2 million pieces of benign flow data and about 550 thousand pieces of malicious behavior flow data containing five major attacks: Denial of Service (DoS) attack, web attack, infiltration attack, port scan attack, and brute-force (BF). CIC-IDS2012 contains about 1.5 million pieces of benign flow data and no more than 50,000 pieces of abnormal flow data with 4 types: brute-force, infiltration, Hyper Text Transfer Protocol (HTTP) DoS, and Distributed Denial of Service (DDoS). At the same time, USTC-TFC2016 merges about 300,000 pieces of benign and 600,000 pieces of 10 types of Trojan horse attack abnormal flow data. As for these abnormal types, DoS attack is an attack that causes denial of service whose purpose is to make the computer or network unable to provide normal services. Web attack often attacks websites or web applications and causes the application not to work normally. Infiltration attack is a systematic and progressive comprehensive attack mode whose attack target is often clear, but the purpose is not so single. Port scan is the common way to search target computer by hackers that can find out some potential intrusion ports. Brute-force attack means that hackers use the password dictionary to guess the user's password by exhaustive method, which is one of the most widely used attack methods. HTTP DoS and DDoS are two special DoS attacks.

4.2. Data Processing. It is described in Section 4.1 that PCAP files are original data of experiments. But the neural network model cannot identify these data types. Traffic packets need to be converted into vectors with the same length. It can be seen from Figure 7 that the data processing covers four steps of data split, data clean, data transfer, and data trimming. Once this process is finished, the data can be sent to the model as an input value.

Data Split: A PCAP file records the network behavior information spanning for a period of time, which means that a file contains a large number of different access connections. In general, people take a combination of source IP address, source port, destination IP address, destination port, and transport protocol as a single access behavior, which is also called 5-tuple. Therefore, it is necessary to divide a large PCAP file into a number of small PCAP files that only contain one 5-tuple data. This small PCAP file is named a traffic flow as well.

Data Clean: Almost all the data generated by practical applications will inevitably have redundancy, and so does network traffic. In this step, empty flows, duplicate flows, and interferential flows are picked out and deleted to produce the available data.

Data Transfer: DL model cannot analyze PCAP files, so they need to be transferred into text files with hexadecimal number as content. This step will not lose any information, just the conversion of the storage form.

Data Trimming: This step will determine the length of hex data. Normally, the length depends on business experience, packets average length, flows average length, and suchlike factors. In this paper, we select 784 bytes of data and reshape them to a 28×28 matrix. If a PCAP file is larger than 784 bytes, it will be trimmed to 784 bytes. Or if it is smaller than 784, 0×00 will be increased into the end of the flow.

4.3. Results and Analysis

4.3.1. Detection Results Output from the Model. As same as in other classification tasks, we collect four evaluation indexes to evaluate the effect of the model, macro-f1, weighted-f1, recall, and precision, which are written as I_{mf1}, I_{wf1}, I_{recall}, and I_{pre} and are explained as follows:

$$I_{\text{recall}} = \frac{1}{N}\sum_{i=1}^{N}\frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{11}$$

$$I_{\text{pre}} = \frac{1}{N}\sum_{i=1}^{N}\frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{12}$$

$$I_{\text{mf1}} = \frac{1}{N}\sum_{i=1}^{N}\frac{2 * I_{\text{pre}} * I_{\text{recall}}}{I_{\text{pre}} + I_{\text{recall}}}, \tag{13}$$

$$I_{\text{wf1}} = \frac{1}{N}\sum_{i=1}^{N} w\frac{2 * I_{\text{pre}} * I_{\text{recall}}}{I_{\text{pre}} + I_{\text{recall}}}, \tag{14}$$

where TP represents the number of correctly identified positive samples, TF represents the number of correctly identified negative samples, FP denotes the number of wrongly identified positive samples, and FN denotes the number of wrongly identified negative samples. N refers to the category number in data and w is the weight of this category to the total data quantity.

Under the experiment, the model results are verified on three datasets described in Section 4.1. At the same time, we set up four groups of comparative experiments on each dataset. In a further step, the training set and test set use the same data to ensure the effectiveness and the reference value of the four comparison experiments shown in Tables 1–3.

(1) Similarity Measure Method. This method exerts the Pearson correlation coefficient to calculate the distance between the test sample and the center samples of benign data in the training set. Only benign data are employed to find the center of the training samples. When calculating, we choose n samples randomly and calculate the nearest center vector to them as the center sample. The abnormal behavior is detected by the similarity distance between the test sample and the normal data center obtained via the training process. This method attempts to use the most primitive data distribution to distinguish outliers. Due to the complex distribution of the original data, it can be predicted that its accuracy will not be ideal.

(2) K-Means Method. K-means method applies the unsupervised clustering to the detection of anomalies directly. The unsupervised process means that a lot of labeled data is not required in this process. K-means is a distance-based algorithm selecting the centroids of each category and comparing the distance between the test samples to each centroid. The sample will eventually be classified into the category with the nearest centroid. Assuming that we have 2 categories, normal and abnormal, there are 2 cluster centers. According to the principle of minimum distance from the initial center point, all observations are divided into the categories, where each center point is located. Then the mean value of the observation points in each category is calculated as the center of the next iteration until convergence. The unsupervised method is always sensitive to data. Because there is a lot of information redundancy in traffic data, it will have a greater impact on the clustering process.

(3) AE Method. As part of the proposed model, AE reconstructs the data directly with the original data resource, the advantage of which is that it needs no abnormal behavior data and simply makes use of the normal traffic data. But the effects are often decided by the original distribution of data.

(4) CNN with AE. CNN with AE also belongs to the proposed model without adding the attention mechanism. Section 3.2 and Figure 5 concretely reveal these treatment processes. Only 2,000 malicious samples produce a marked effect in the supervised classifier of each dataset, by virtue of which the unsupervised autoencoder can reach a state of

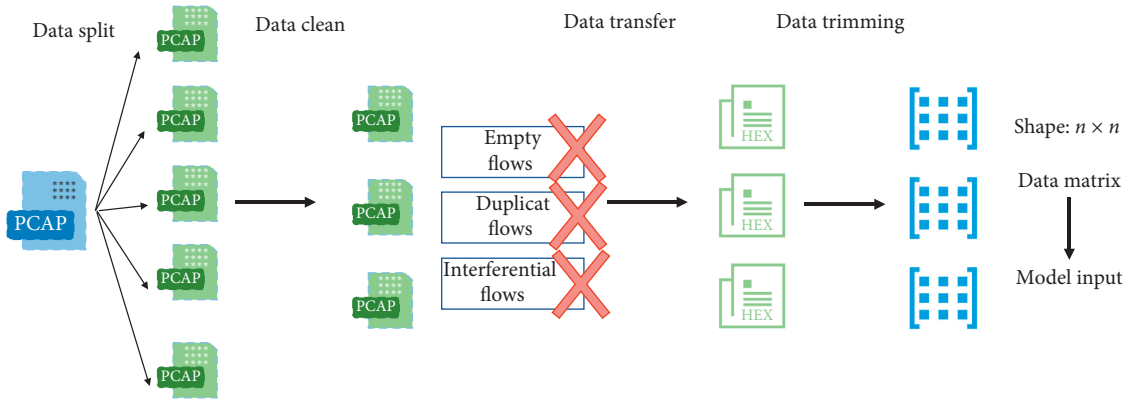


Figure 7: Data processing outputs data matrixes with shape n.

Table 1: Experiments results of dataset CIC-IDS2017.

Methods	With unknown categories			Known categories		
	Recall	Precision	Macro-f1	Recall	Precision	Macro-f1
Similarity measure	0.5233	0.5134	0.5103	0.4134	0.4230	0.4071
K-means cluster	0.6428	0.5554	0.3228	0.6163	0.6962	0.4721
AE	0.6226	0.5929	0.6042	0.8510	0.8148	0.8196
CNN with AE	0.6718	0.6399	0.6539	0.8626	0.8262	0.8323
DFAE	**0.8593**	**0.8738**	**0.8579**	**0.9519**	**0.9508**	**0.9513**

perfection. More than 80,000 test samples are put into use, including 30,000 benign samples and other abnormal ones.

Tables 1–3 present the experiment results among three datasets. Each method adopts two types of test data. One only contains samples of known categories trained in the surprised classifier and the other also embodies quite a lot of unknown categories. It can be concluded that there is an obvious improvement in every dataset with the proposed model. It is proved that the pretraining with a small number of malicious samples can significantly improve the subsequent data reconstruction process. As for USTC-TFC2016 and CIC-IDS2012, the supervised training process enhances the recall, precision, and f1-score over 10%. The improvement of CIC-IDS2017 is lightly inferior to others. Meanwhile, the effect of the attention mechanism on CIC-IDS2017 is more obvious by reforming the results over 10%, especially on the test samples with unknown categories. From the results of three datasets, we can see that the proposed method is very effective. As for better distributed dataset such as USTC-TFC2016, the detection accuracy is more than 99%. As for the dataset with general experimental results such as CIC-IDS2017, its accuracy also reached 95%. This also shows that the model has made a significant improvement on different distribution of data. The proposed method has strong generalization ability and provides the possibility for the application in the actual data.

4.3.2. Some Other Discussion. Apart from the enhancement of the overall effect, we also observed the classification of normal and abnormal samples. Figure 8 manifests the confusion matrix of the results output from DFAE on each dataset. In view of the effect of each category, the detection accuracy of the normal flow is better than that of the abnormal ones. From the perspective of confusion matrix results, the accuracy is relatively balanced between positive and negative samples. It is particularly valuable to achieve this result in few-shot learning tasks. Our experiment emerged the imbalance that the number of normal samples is much larger than abnormal ones. However, the proposed model can well solve this problem. The findings suggest that the detection rate performs efficaciously in both categories.

In order to investigate the effect of the model more deeply, we analyze the reconstruction results of the original data with and without the deep features. From Figure 9, we can see the reconstruction rate of different datasets. On the red arrow's left side is the data reconstruction of original data based on AE. On the right side of the red arrow is the data reconstruction process using the deep features output from a supervised classifier based on CNN. It can be easily discovered that the reconstruction rate of the right figure is obviously better than that of the left one. There is a clear reconstruction rate between normal samples and abnormal samples in terms of the right figure. Every row in the graph is the result of a dataset. In each dataset, 1,000 samples are randomly picked up to compare the reconstruction result in the figure. The green rectangles represent the rate of direct reconstruction using the original samples and the blue ones denote the reconstruction rate of deep-features-based results. We can see an apparent dividing line between green and blue in the figure to the right of the red arrow. It can be safely concluded that the proposed method has verified the feasibility of the model in the small sample anomaly detection task from both theory and effect.

Table 2: Experiments results of dataset CIC-IDS2012.

Methods	With unknown categories			Known categories		
	Recall	Precision	Macro-f1	Recall	Precision	Macro-f1
Similarity measure	0.5545	0.5567	0.5555	0.4996	0.4996	0.4996
K-means cluster	0.2394	0.3367	0.2798	0.4595	0.3238	0.3799
AE	0.8066	0.8713	0.8335	0.7908	0.8372	0.8068
CNN with AE	0.9455	0.9522	0.9488	0.9418	0.9519	0.9465
DFAE	**0.9490**	**0.9668**	**0.9576**	**0.9480**	**0.9649**	**0.9557**

Table 3: Experiments results of dataset USTC-TFC2016.

Methods	With unknown categories			Known categories		
	Recall	Precision	Macro-f1	Recall	Precision	Macro-f1
Similarity measure	0.4525	0.4502	0.4514	0.2693	0.2630	0.2660
K-means cluster	0.7965	0.6003	0.5557	0.6535	0.6861	0.5465
AE	0.8105	0.9654	0.8692	0.8573	0.9050	0.8751
CNN with AE	0.8765	0.8410	0.8576	0.9565	0.9675	0.9617
DFAE	**0.9552**	**0.9675**	**0.9613**	**0.9983**	**0.9965**	**0.9973**

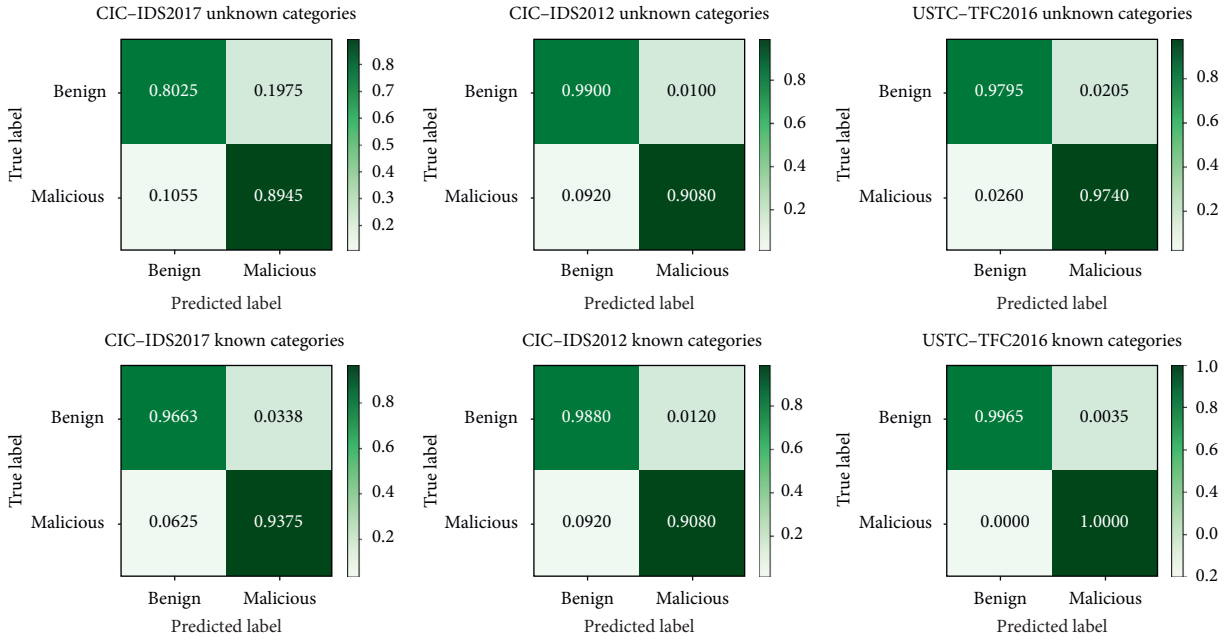


Figure 8: Confusion matrix of experiment results on each dataset.

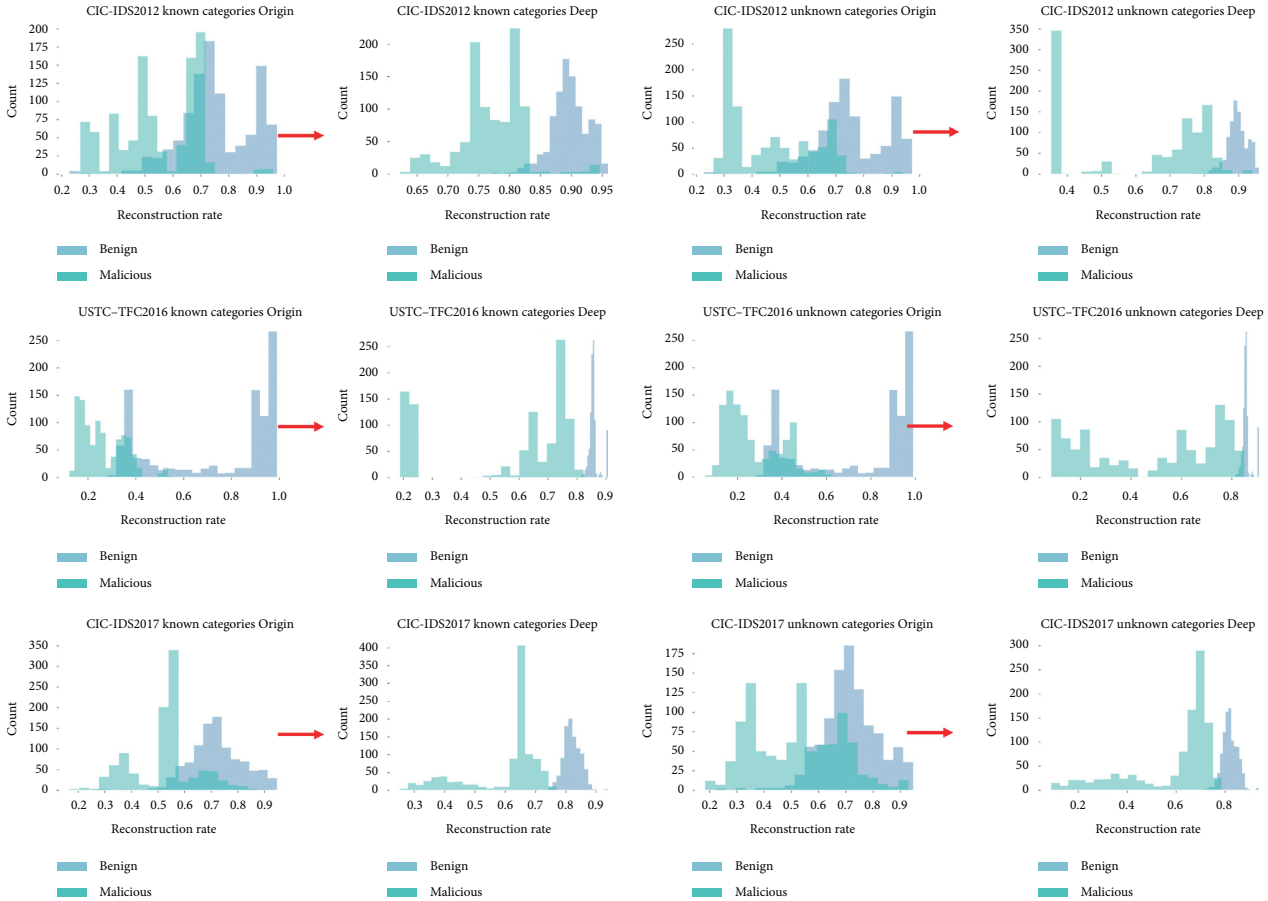


FIGURE 9: Reconstruction rate of different datasets. The image on the left of the red arrow is the result of reconstruction using AE only and the image on the right of the red arrow is the result of data reconstruction using the proposed method.

5. Conclusion

This paper proposes an anomaly detection model with a few abnormal samples to solve the problem in few-shot detection based on CNN and AE. This is a very common scenario in the application of network abnormal behavior detection. In the actual production and application, the number of normal behaviors is far greater than the number of abnormal behaviors. The model proposed in this paper can solve this problem and improve the detection results. As described above, the results demonstrate the improvements under this module of detection recall, precision, and f1-score in three datasets. What is more, the experiment proves that, through a small number of malicious samples for pretraining, the data reconstruction task will become easier, and the few-shot detection effect can also be improved in an obvious way. Sufficient comparative experiments verified the effectiveness of the proposed method. With regard to the network traffic detection, it is not easy to detect the anomaly directly through an unsupervised model, which can be easily reflected from comparative experiments. Therefore, it is necessary to deal with the limited amount of labeled abnormal behavior samples. The few-shot malicious traffic detection task plays a crucial role in practical applications.

In the foreseeable future, we will continue to raise the few-shot malicious traffic detection results by different methods and to increase the possibility of applying this model into the actual and real-time network traffic. As for actual network traffic anomaly detection, it is always difficult to find out malicious behaviors when a series of new attacks occur. It is unrealistic to focus on training and learning new categories. But few-shot detection is meaningful here. Most of the time, we are concerned about the normal data and the new types of abnormal samples are only trained in the set intervals by few-shot learning. This is also the reason why this method is more feasible in the actual application scenarios. Meanwhile, we will tend to utilize fewer training samples to achieve a higher outcome on different datasets and production data. Detection efficiency will also be one of the focuses in future work.

Data Availability

The datasets used in this paper are available at https://www.unb.ca/cic/datasets/, and they are also available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The authors acknowledge ICN&CAD Laboratory of School of Electronic Engineering, Beijing University of Posts and Telecommunications, for the experimental environment. Special thanks are due to Lv Tianqi, Xiao Yabo, Wu Lingrui, Fan Yiqi, Keiven, and Zhang Yu for their great help in this work. This work was supported by the National Natural Science Foundation of China (Grant no. 62071056).

References

[1] Statista Research Department, "IoT: number of connected devices worldwide 2012–2025, Statista," 2019, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[2] S. A. R. Shah and I. Biju, "Performance comparison of intrusion detection systems and application of machine learning to snort system," *Future Generation Computer Systems*, vol. 80, no. 1, pp. 157–170, 2017.

[3] A. Beaugnon and P. Chifflier, "Machine learning for computer security detection systems: practical feedback and solutions," 2018, https://www.ssi.gouv.fr/uploads/2018/11/machine-learning-for-computer-security-abeaugnon-pchifflier-anssi-.pdf.

[4] K. Sailesh and S. Dharmapurikar, "Algorithms to accelerate multiple regular expressions matching for deep packet inspection," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 339–350, 2006.

[5] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2014.

[6] Z. G. Qu, S. Y. Chen, and X. J. Wang, "A secure controlled quantum image steganography algorithm," *Quantum Information Processing*, vol. 19, no. 380, pp. 1–25, 2020.

[7] C. Qian, X. Li, N. Sun, and Y. Tian, "Data security defense and algorithm for edge computing based on mean field game," *Journal of Cyber Security*, vol. 2, no. 2, pp. 97–106, 2020.

[8] D. Plonka, "FlowScan: a network traffic flow reporting and visualization tool," in *Proceedings of the 14th Conference on Systems Administration (LISA 2000)*, New Orleans, LA, USA, December 2000.

[9] E. Kenigsberg, M. Gopshtein, and N. Ioffe, "Collecting network-level packets into a data structure in response to an abnormal condition," 2012, https://www.freepatentsonline.com/8135979.pdf.

[10] J. Liu, F. Liu, and N. Ansari, "Monitoring and analyzing big traffic data of a large-scale cellular network with Hadoop," *IEEE Network*, vol. 28, no. 4, pp. 32–39, 2014.

[11] M. Z. N. Saavedra and W. E. S. Yu, "A comparison between text, parquet, and PCAP formats for use in distributed network flow analysis on Hadoop," *International Conference on Networking and Information Technology*, vol. 5, no. 2, pp. 59–64, 2017.

[12] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proceedings of the 2017 International Conference on Information Networking (ICOIN)*, pp. 712–717, Da Nang, Vietnam, January 2017.

[13] R. Bala and R. Nagpal, "A review on KDD CUP99 and NSL-KDD dataset," *International Journal of Advanced Research in Computer Science*, vol. 10, no. 2, p. 64, 2019.

[14] L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[15] P. Negandhi, Y. Trivedi, and R. Mangrulkar, "Intrusion detection system using random forest on the NSL-KDD dataset," in *Proceedings of the Emerging Research in Computing, Information, Communication and Applications*, pp. 519–531, Singapore, 2019.

[16] I. H. Sarker and A. S. M. Kayes, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big Data*, vol. 7, no. 1, pp. 1–29, 2020.

[17] B. G. Atli, Y. Miche, A. Kalliola, I. Oliver, S. Holtmanns, and A. Lendasse, "Anomaly-based intrusion detection using extreme learning machine and aggregation of network traffic statistics in probability space," *Cognitive Computation*, vol. 10, no. 5, pp. 848–863, 2018.

[18] L. D'Hooge and A. S. M. Kayes, "Inter-dataset generalization strength of supervised machine learning methods for intrusion detection," *Journal of Information Security and Applications*, vol. 54, no. 1, p. 102564, 2020.

[19] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "Deep-Full-Range: a deep learning based network encrypted traffic classification and intrusion detection framework," *IEEE Access*, vol. 7, no. 1, pp. 45182–45190, 2019.

[20] N. Thapa, Z. Liu, D. B. Kc, B. Gokaraju, and K. Roy, "Comparison of machine learning and deep learning models for network intrusion detection systems," *Future Internet*, vol. 12, no. 10, p. 167, 2020.

[21] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Intrusion detection evaluation dataset (CIC-IDS2017)," in *Proceedings of the of Canadian Institute for Cybersecurity*, Fredericton, Canada, 2018.

[22] A. Javaid, Q. Niyaz, and W. Sun, "A deep learning approach for network intrusion detection system," in *Proceedings of the the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, pp. 21–26, New York City, NY, USA, December 2016.

[23] M. Chen, X. Wang, M. He, L. Jin, K. Javeed, and X. Wang, "A network traffic classification model based on metric learning," *CMC-computers Materials & Continua*, vol. 64, no. 2, pp. 941–959, 2020.

[24] M. Song, J. Ran, and S. Li, "Encrypted traffic classification based on text convolution neural networks," in *Proceedings of the IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, pp. 432–436, Dalian, China, October 2019.

[25] H. Zhu, F. Meng, S. Rho et al., "Long short term memory networks based anomaly detection for KPIs," *Computers, Materials & Continua*, vol. 61, no. 2, pp. 829–847, 2019.

[26] C. Du, S. Liu, L. Si, Y. Guo, and T. Jin, "Using object detection network for malware detection and identification in network

traffic packets," *Computers, Materials and Continua*, vol. 64, no. 3, pp. 1785–1796, 2020.

[27] Y. Wei, F. R. Yu, M. Song et al., "User scheduling and resource allocation in HetNets with hybrid energy supply: an actor-critic reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 680–692, 2017.

[28] Y. Wei, F. R. Yu, M. Song et al., "Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor–critic deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2061–2073, 2018.

[29] G. Caminero, M. Lopez-Martin, and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Computer Networks*, vol. 159, no. 1, pp. 96–109, 2019.

[30] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, no. 1, pp. 249–259, 2018.

[31] S. Rodda and U. S. R. Erothi, "Class imbalance problem in the network intrusion detection systems," in *Proceedings of the 2016 international conference on electrical, electronics, and optimization techniques (ICEEOT)*, pp. 2658–2688, Chennai, India, March 2016.

[32] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-supervised *K*-means DDoS detection method using hybrid feature selection algorithm," *IEEE Access*, vol. 7, no. 1, pp. 64351–64365, 2019.

[33] H. Xu, W. Chen, N. Zhao et al., "Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications," in *Proceedings of the 2018 World Wide Web Conference*, pp. 187–196, Lyon, France, April 2018.

[34] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[35] D. Yuan, K. Ota, M. Dong et al., "Intrusion detection for smart home security based on data augmentation with edge computing," in *Proceedings of the 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, Dublin, Ireland, June 2020.

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[37] Y. Xiao, D. Yu, X. Wang et al., "SPCNet: spatial preserve and content-aware network for human pose estimation," in *Proceedings of the 24th European Conference on Artificial Intelligence*, pp. 2776–2783, Santiago de Compostela, Spain, August 2020.

[38] C. S. N. Pathirage, J. Li, L. Li, H. Hao, W. Liu, and R. Wang, "Development and application of a deep learning-based sparse autoencoder framework for structural damage identification," *Structural Health Monitoring*, vol. 18, no. 1, pp. 103–122, 2019.

[39] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal*, pp. 8609–8613, Vancouver, Canada, May 2013.

[40] M. Lau and K. Lim, "Investigation of activation functions in deep belief network," in *Proceedings of the 2nd International Conference on Control and Robotics Engineering (ICCRE)*, pp. 201–206, Vancouver, Canada, April 2017.

[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Salt Lake City, UT, USA, June 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6682674, 9 pages
https://doi.org/10.1155/2021/6682674

Research Article

Hardware Trojan Detection Based on Ordered Mixed Feature GEP

Huan Zhang,[1,2] **Jiliu Zhou**,[1] **Dongrui Gao**,[2,3] **Xinguo Wang**,[2] **Zhefan Chen**,[4] **and Hongyu Wang**[2]

[1]*College of Computer Science, Sichuan University, Chengdu 610065, China*
[2]*School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China*
[3]*Center for Information in Biomedicine, School of Life Sciences and Technology, University of Electronic Science and Technology of China, Chengdu 611731, China*
[4]*Faculty of Science, Simon Fraser University, Burnaby V5A 1S6, Canada*

Correspondence should be addressed to Jiliu Zhou; zhoujl@cuit.edu.cn

Received 28 December 2020; Revised 24 January 2021; Accepted 21 February 2021; Published 18 March 2021

Academic Editor: Liguo Zhang

In the hardware Trojan detection field, destructive reverse engineering and bypass detection are both important methods. This paper proposed an evolutionary algorithm called Ordered Mixed Feature GEP (OMF-GEP), trying to restore the circuit structure only by using the bypass information. This algorithm was developed from the basic GEP through three sets of experiments at different stages. To solve the problem, this paper transformed the GEP by introducing mixed features, ordered genes, and superchromosomes. And the experiment results show that the algorithm is effective.

1. Introduction

At all stages of the life cycle of integrated circuits (IC), there are security vulnerabilities for hardware in the global business model of semiconductor supply chain. In the current hardware Trojan detection technology, destructive reverse engineering [1–3] is good but costly bypass detection [4–9] is the technology whose cost is low, which is a development key direction at present.

An evolutionary algorithm called Ordered Mixed Feature GEP (OMF-GEP) is proposed in this paper. This algorithm takes a single-circuit component as a node to form a mixed feature of various logical or physical features of the node. And it can find the original circuit by using the GEP function regression ability.

2. Mixed Features

The logic value of a circuit or any kind of bypass information such as voltage and current can be considered as a manifestation of a characteristic of the circuit. When only one characteristic representation value is used to represent the circuit, if other constraints are not added, there will undoubtedly be a variety of circuits to meet the requirements of this single feature. The two circuits are shown in Figure 1.

If you look only at the logical values, the two circuits are completely equivalent, and both of which are

$$Y = CD. \tag{1}$$

You can also see that in the circuit in Figure 1(b), inputs A and B are used at all; that not, the input s A and B in the first circuit do not actually affect the output.

This example is only the value of the circuit logic value and the circuit bypass information detection, and there is a similar situation. Multiple different circuit structures can be obtained for the detection results of any single bypass information. It can be seen that only using logical values or bypass information to describe the circuit will lead to too much isomorphism to confirm the circuit structure.

The essence of hardware Trojan horse design is to add additional circuit to normal circuits, but the performance of the whole circuit on some features (the most common is the logic value) is the same as that of the normal circuit, to realize hiding. However, this additional circuit will inevitably cause other circuit features to change.

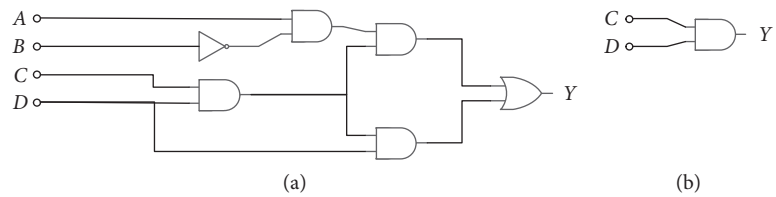


Figure 1: Two circuits with the same logical value.

In this paper, the logic value of the circuit or any kind of bypass information such as voltage and current is called a feature. For the isomorphism of a feature, it is essentially due to the superposition of the features of the circuit elements on the feature. The features of multiple different circuit structures with the feature are similar or even the same, so that the corresponding circuit cannot be represented by the result of a feature.

Then, when detecting multiple features at the same time, multiple isomorphic circuits can be obtained from the detection results of each feature, but the superposition features of different features cannot be exactly the same. These isomorphic circuits cannot be the same, where the same part is a possible real circuit.

Figure 2 illustrates the application of the algorithm to a diode-designed And gated circuit.

Its logical meaning is

$$Y = A \cdot B. \tag{2}$$

It has a lot of physical meaning. Here is description of its voltage:

$$V_y = \min\left(V_A, V_B\right) + V_{\text{dio}}. \tag{3}$$

Among them, V_y represents voltage of the output position Y; V_A and V_B represent the voltage values of two input locations A and B; V_{dio} represents the diode conduction voltage for silicon tubes, and its value is often 0.7.

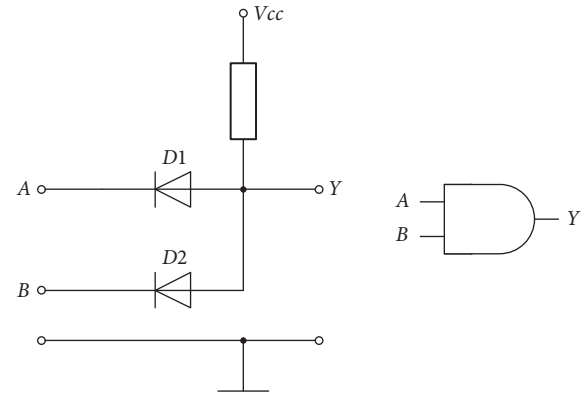


Figure 2: And gate circuits.

Then, the And gate can be expressed as

$$\left(y_{\text{logic}}, y_v\right) = G_{\text{and}}\left(x_{1\_\text{logic}}, x_{2\_\text{logic}}, x_{1\_v}, x_{2\_v}\right). \tag{4}$$

Then, a measurement of k features, n input, and single-output single-gate circuit can be expressed as Expression 1:

$$\left(y_1, y_2, \ldots, y_k\right) = G\left(\left(x_{1,1}, x_{1,2}, \ldots, x_{1,n}\right), \left(x_{2,1}, x_{2,2}, \ldots, x_{2,n}\right), \ldots, \left(x_{k,1}, x_{k,2}, \ldots, x_{k,n}\right)\right). \tag{5}$$

Among them, $y_i\,(i = 1, \ldots k)$ is the output values for the adoption of the kth feature. $x_{i,j}\,(i = 1, \ldots, k, j = 1, \ldots n)$ is the ith input value for the adoption of the feature j.

Without losing generality, let us define

$$\begin{aligned} Y &= \left[y_1, y_2, \ldots y_k\right], \\ X_i &= \left[x_{1,i}, x_{2,i}, \ldots, x_{n,i}\right]^T, \quad i = 1, 2, \ldots, k, \\ X &= \left[X_1, X_2, \ldots, X_k\right]. \end{aligned} \tag{6}$$

Then, Expression 1 can be expressed as

$$Y_{1*k} = G\left(X_{n*k}\right), \tag{7}$$

which is Expression 2.

3. Algorithm 1: Single-Output Circuit

3.1. GEP Representation. In recent years, there have been many studies based on evolutionary algorithms and multisource data such as data fusion of adaptive weighted multisource sensor [10], the research on evolutionary algorithm for symbolic network [11], the application of the genetic algorithm in multiobjective multicast routing [12], and multiplicity problems in genetic association studies [13]. Zhi and Liu [14] proposed a new GA algorithm for mechanical design optimization problems. These studies gave us the inspiration to use the evolutionary algorithms in the hardware Trojan detection.

Gene expression programming (GEP) [15] is an evolutionary computing algorithm that has performed well in the study of evolutionary hardware [16–22]. It can solve the problem of tree structure very well. For the multi-input/single-output tree structure circuit, it can be described as a tree with n leaf nodes, which can be represented directly by GEP. As shown in Figure 3, the 6-input/1-output logic circuit can be easily represented as a tree structure, in which the logic gate function is replaced by the logic symbol, and the corresponding effective gene is

$$\text{And, And, And, } A\text{, Not, And, And, } B, C\text{, Not, } E, F, D. \quad (8)$$

3.2. Algorithm of Mixed Feature GEP. One operator in GEP represents only one kind of calculation, and a GEP individual can only represent one test item, so the idea of algorithm one is to merge the multiple test results of a basic circuit into a function expression. Combined into a compound function, that is, let a function represent multiple calculations and evolve a representation close to the original circuit. Specifically, these multiple detection values are included in a function, the input of the function is multiple values, and the output result is a vector, such as the aforementioned gate circuit, which is still represented as "And" in the GEP expression tree. However, its meaning has become the following vector calculation:

$$\text{And}(A_1, A_2, \ldots A_n) = [F_1(A_1), F_2(A_2), \ldots, F_n(A_n)]. \quad (9)$$

The $A_k = (k = 1, \ldots n)$ is the input value of a detection, and $F_k(A_k)(k = 1, \ldots n)$ is the result of this detection A_k to the input value.

For example, for this gate circuit, the symbol And means the following:

$$\text{And}((L_A, L_B), (V_A, V_B), (C_A, C_B)) = [L_A L_B, \min(V_A, V_B) + V_{\text{dio}}, C_A + C_B]. \quad (10)$$

Among them, L_A, L_B represent the logical value (1 or 0) of voltage input of the A or B point, V_A, V_B represent the input voltage of the A or B point, and C_A, C_B represent the input current of the A or B point.

Thus, when using GEP evolution, a symbolic value can simultaneously represent multiple unrelated items. This algorithm will be called Mixed Feature GEP (MF-GEP).

3.3. Experiment Setup. The experiment is limited to the use of simple logic gate circuits, does not involve triggers, clocks, etc., and does not consider time effects.

Four groups of experiments were designed.

Output $m = 1$,

The number of features are $k = 1, 2, 2, 3$.

Three features are used: feature 1 is the logical value, feature 2 is the voltage value, and feature 3 is the current value.

As a comparative experiment, the parameters used are identical as Table 1 shows.

3.4. Design of Fitness Function. The feature data are logic data, voltage data, and current data, which have their own fitness.

Logical data fitness is

$$F_{\text{logic}} = 1 - \frac{\sum_{i=1}^{N} |y_i - y|}{N}, \quad (11)$$

which is Expression 3.

N is the number of test data, y_i is the logic value calculated according to the test data after decoding, and y is the output logic value of the test data. Because it is a logical value, the worst case is that each output decoded by the individual is opposite to the test value, that is, $|y_i - y| = 1$, so F_{logic} is among the range of $[0, 1]$.

Voltage data fitness is

$$F_{\text{vol}} = 1 - \frac{\left(\sum_{i=1}^{N} |y_i - y|/N\right)}{(V_{\text{CC}} - V_{\text{DD}})}, \quad (12)$$

which is Expression 4.

N is the number of test data, y_i is the voltage value calculated according to the test data after decoding, and y is the output voltage value of the test data. At worst, each test output value is either the highest level or the lowest level, and each output decoded by the individual is opposite to the test value $|y_i - y| = V_{\text{CC}} - V_{\text{DD}}$; therefore, F_{vol} is among the range of $[0, 1]$.

Current data fitness is

$$F_{\text{cir}} = 1 - \frac{\text{SSE}}{\text{SST}}, \quad (13)$$

which is Expression 5.

Among them,

$$\text{SSE} = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2,$$
$$\text{SST} = \sum_{i=1}^{m} (y_i - \overline{y}_i)^2, \quad (14)$$

where y_i is the data observation, $\hat{y}_i$ is the estimate value of the y_i which is calculated from the decoding expression, and $\overline{y}_i$ is the average value of the variable y. That is, the SSE is the Sum of Squared Errors and the SST is the Sum of Squares in Total. F_{cir} the square of the multicorrelation coefficient in statistics.

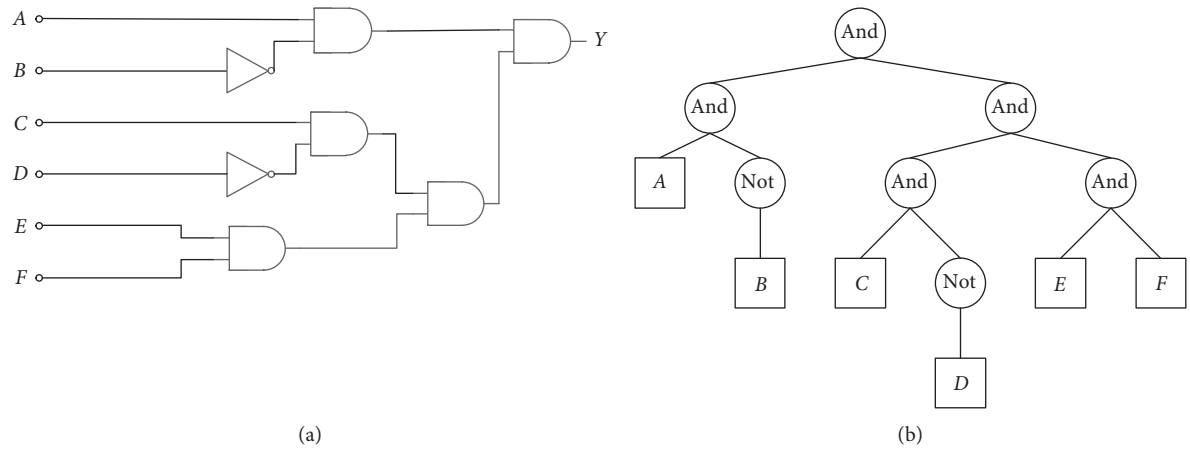


Figure 3: A single-output circuit and its expression tree.

Table 1: Parameter settings for experiment 1.

Parameter	Value
Stop	Fitness = 1
Selection mode	Tournament, size = 3
Population size	10000
Head length	20
Tail length	21
Chromosome length	1
Mutation rate	0.05
Insert rate	0.1
Root insert rate	0.01
One-point cross rate	0.1
Two-point cross rate	0.1
Input number	4
Output number	1
Function set	Not, and, or

According to the previous algorithm description, the individual fitness should be a combination of the three; then, the individual fitness is

$$F = C_1 \cdot F_{\text{logic}} + C_2 \cdot F_{\text{vol}} + C_3 \cdot F_{\text{cir}}, \quad \left(\sum C_k = 1\right), \quad (15)$$

which is Expression 6: individual fitness expression.

C_1, C_2, C_3 are the weight of three features in the final fitness.

3.5. Experiment. Figure 4 shows a circuit. Its Boolean expression is

$$Y = A + BCD' + BC'D. \quad (16)$$

Its calculation is

$$Y = \begin{cases} 0, & (ABCD) < (0101)_2 \text{ or } (ABCD) = (0111)_2. \\ 1, & \text{else.} \end{cases} \quad (17)$$

The input value $(ABCD) = (0111)_2$ can be seen as the Trojan trigger conditions. When there are more pins, only part of the value can be tested, and you may miss the input $(ABCD) = (0111)_2$. In the following experiment, the input data will not provide the (0111) value to trigger the Trojan, and the output of the input value will be determined by the evolved circuit.

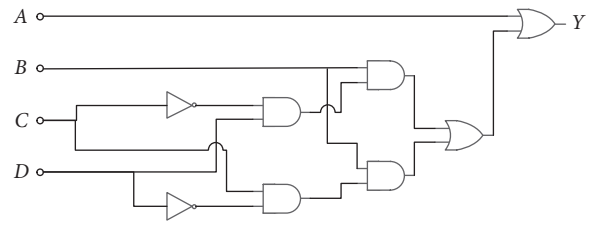


Figure 4: A circuit with a Trojan.

Its effective gene is

$$\text{Or}, A, \text{Or}, \text{And}, \text{And}, B, \text{And}, B, \text{And}, \text{Not}, D, C, \text{Not}, C, D. \quad (18)$$

Using different combination forms, we designed 4 groups of experiments. Considering that the logic value is required to be correct first in the circuit, the voltage value and the current value must be meaningful on the basis of the correct logic value, so the logic value is included in each group of experiments, and the fitness of the individual combines several data; the logical value accounts for a larger proportion. Table 2 shows the results of the experiments.

The experimental results show the following:

(1) Only using a single feature cannot find Trojan circuit.

(2) Using multiple features can effectively discover Trojan circuits.

(3) The features with direct correlation have no effect on the discovery probability of Trojan horse: in experiment 2, two features of logic value and voltage are used at the same time, and the Trojan horse cannot be found; in experiment 4, although three features are used, the probability of finding Trojan horse is not higher than that of real 3. The reason is that in digital circuits, the logical value itself is expressed by the voltage value; for example, the voltage value less than 3 V is considered 0, and the

Table 2: Comparison of results of 4 groups in experiment 1.

Parameter	Exp1Value	Exp2Value	Exp3Value	Exp4Value
Logic gate	And, or, not	And, or, not	And, or, not	And, or, not
Values provided	Logic values	Logic values Voltage values	Logic values Current values	Logic values Voltage values Current values
Fitness function	$F = F_1$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_2$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_3$	$F = 0.6 \cdot F_1 + 0.2 \cdot F_2 + 0.2 \cdot F_3$
Exercise count	100			
Trojan discovered count	0	0	72	67

voltage value greater than 3 V is considered 1. Therefore, there is no difference between the logic value and voltage value.

4. Algorithm 2: Multioutput Circuit

4.1. GEP Representation. One circuit n input/m output can be described as a forest composed of m trees, each with $1 \sim n$ leaf nodes. The 6-input/2-output circuit in Figure 5 can be decomposed into two tree structured multi-input/single-output circuits.

The circuits shown in Figure 6 can be divided into two independent multi-input/single-output.

The corresponding effective genes are

$$\text{And, And, And, } A\text{, Not, And, And, } B, C\text{, Not, } E, F, D. \quad (19)$$

$$\text{Or, And, And, And, And, And, } F, C\text{, Not, } E, F, E, F, D. \quad (20)$$

The combination of the two genes represents a 6-input/2-output circuit.

4.2. Algorithm of Ordered Mixed Feature GEP. The GEP should be modified as the following to be able to represent this kind of circuit.

4.2.1. Remove the Link Function and Number the Gene. The GEP data structure has its own multigene structure. In formula mining, the basic idea of GEP is to use a polynomial approximation method, so that each independent gene can evolve a part of the final polynomial and then use a connection function (usually "+") to form a complete polynomial. Of course, if the test data are error-free and the cost is sufficient, GEP final expression does not need to be approximated, and it is the expression from which the test data themself come.

The operator such as "+" has a characteristic that there is no sequential difference between the operators. If such an operator is used, it can be considered that there is no sequential difference between the genes in GEP chromosome.

We can also see that GEP can solve a problem similar to $y = f(x_1, x_2, \ldots, x_n)$ function problem; that is, it can deal with the problem of multi-input and single-output. However, circuit combinations are often a multi-input/multi-output problem, that is, a problem as $(y_1, y_2, \ldots, y_m) = f(x_1, x_2, \ldots, x_n)$. This is a situation GEP cannot handle by its own algorithm.

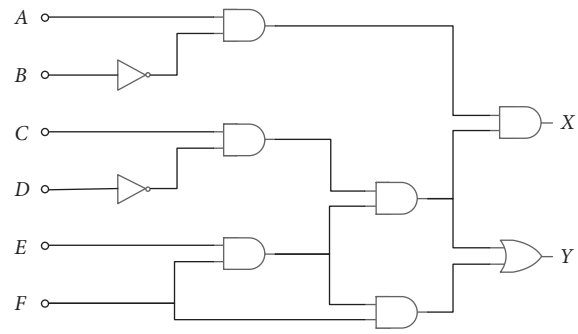


Figure 5: A 6-input/2-output circuit.

For solving the multioutput situation of combinational circuits, the GEP data structure is changed as follows:

(1) The connection function used to connect GEP to multiple genes is removed, so that a gene represents an output, and there is no association between genes; then, a chromosome with k genes represents a circuit with k outputs.

(2) According to the position number of the gene in the chromosome and the position of the gene in the chromosome, the corresponding output pin is represented; that is, the input value in the GEP is the test value of each input pin. The decoding result represents the circuit structure of an output pin. Each gene within a chromosome evolves independently.

4.2.2. Record the Fitness of Each Gene. The fitness is set for each individual in the GEP, which is the basis for the calculation of various evolutionary variations. The fitness represents the approximate degree of the target on the whole of an individual. This fitness is calculated based on the expression tree decoded by an individual.

In the work of this paper, because each gene in an individual is independent of each other, the whole individual decodes not an expression tree, but an expression forest, and the trees in this forest are still orderly. The fitness of an individual depends on each gene. To solve this problem, the fitness is set for each gene of the individual in the work of this paper. The fitness represents the similarity of the gene to the circuit structure of the corresponding pin. Combined with the fitness of all genes, an individual's fitness is formed, indicating the approximation of the individual to the whole circuit structure. Therefore, this paper not only sets the fitness for each individual but also sets the fitness for each gene.

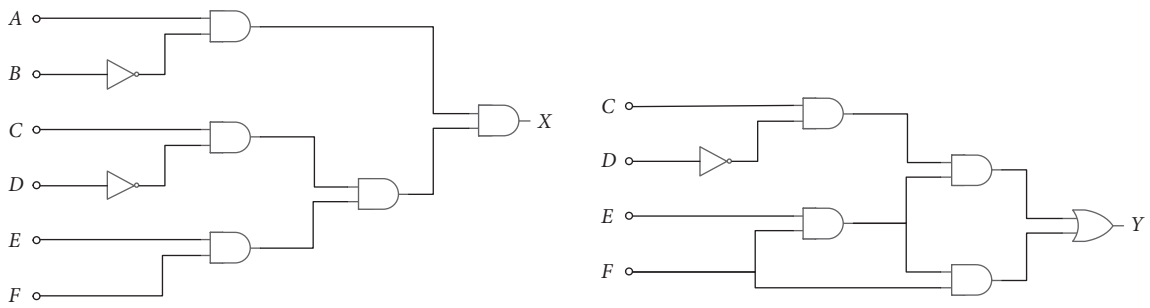


Figure 6: Two circuits divided by the circuit shown in Figure 5.

4.3. Materials and Methods. The relationship between chromosome fitness and gene fitness can be described as

$$\text{fitness}(\text{chromosome}) = \frac{(\sum \text{Fitness}(\text{gene})n)}{n}, \quad (21)$$

which is Expression 7: individual fitness of multigene GEP.

This algorithm is Ordered Mixed Feature GEP (OMF-GEP).

4.4. Experiment Setup. The setting of experiment 2 is most consistent with that of experiment 1. The difference is that the number of genes is increased to 2, corresponding to the operation of gene recombination (the probability is 0.01), and the termination condition of the algorithm is changed to 100,000 times.

The fitness of each gene is calculated in the same way as experiment 1. Expression 5 is used to calculate the fitness of the whole chromosome, in which n is 2.

4.5. Experiment. Use the circuit of Figure 5. In this circuit, Boolean expression is

$$X = AB'CD'EF, Y = EF. \quad (22)$$

Among them,

$$X = \begin{cases} 1, & (ABCDEF) < (101011)_2, \\ 0, & \text{else}. \end{cases} \quad (23)$$

During the experiment, we deliberately hide the test cases that allow X to take a value of 1. The setup of the 4 groups of experiments is completely consistent with that of experiment 1. Table 3 shows the results.

It can be seen from the experimental results that no matter what combination of features is used, after evolution begins, the fitness cannot continue to grow after reaching a very low value, and evolution has actually stopped. The overall trend is shown in Figure 7.

5. Algorithm 3: Superchromosome

In experiment 2, it is impossible to evolve continuously when the fitness is not high in the early stage of evolution. By analyzing the reasons, the fitness of the individual represents the approximate degree of the individual to the whole circuit, but in the design of the modified algorithm, each gene evolves alone. That is, the approximation of the circuit is divided into different parts. In the process of evolutionary calculation, such individuals will be considered poor individuals, with less chance of heredity in the next evolution, resulting in the loss of local genes already leading in evolution. This situation will lead to the efficiency of the algorithm evolution being very inefficient, or even unable to converge, because the evolution has entered a situation of almost random evolution.

5.1. Algorithm Description. To solve this situation, this paper introduces the concept of "superchromosome" in its work. Each individual in the population is obtained by genetic variation after initialization, but the superindividual is artificially constructed. Using the fitness of each gene that has been recorded, a superindividual is constructed after an evolution. The method is that the structure of the superindividual and the ordinary chromosome is the same, but the gene at each position is the best one in the same position in the whole population, as shown below. In this way, the superindividual concentrates the last evolutionary optimal gene at each gene location, and there is no doubt that the superindividual is the optimal individual in the population. Then, replacing the worst individuals in the population with such superchromosome to continue the later evolution can effectively avoid the elimination of local excellent genes. Figure 8 shows how to compose the superchromosome.

The introduction of the superchromosome was intended to avoid the elimination of excellent genes in the same individual due to the existence of "low quality" genes, but it brought an additional benefit. Evolutionary individual fitness changes can often reach a high level soon after evolution, as Figure 9 shows. The reason is that the superindividual concentrates the optimal genes in each gene position, so that the whole individual can achieve very high fitness.

5.2. Experimental Setup. The setting of experiment 3 is the same as that of experiment 2, but it increases the generation of superindividuals when each generation evolves. The fitness function designed is exactly as the same as experiment 2.

5.3. Experiment. The content of the experiment is the same as that of experiment 2. Table 4 shows the results.

The experimental results show the following:

Table 3: Comparison of results of 4 groups in experiment 2.

Parameter	Exp1Value	Exp2Value	Exp3Value	Exp4Value
Logic gate	And, or, not	And, or, not	And, or, not	And, or, not
Values provided	Logic values	Logic values Voltage values	Logic values Current values	Logic values Voltage values Current values
Gene fitness function	$F = F_1$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_2$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_3$	$F = 0.6 \cdot F_1 + 0.2 \cdot F_2 + 0.2 \cdot F_3$
Chromosome fitness	$F_{\text{chrom}} = (F_{\text{gene1}} + F_{\text{gene2}})/2$			
Evolution number	100000			
Max chrom-fitness	0.42	0.46	0.44	0.44
Max evolution no. when fitness stopped advance	92	134	112	137

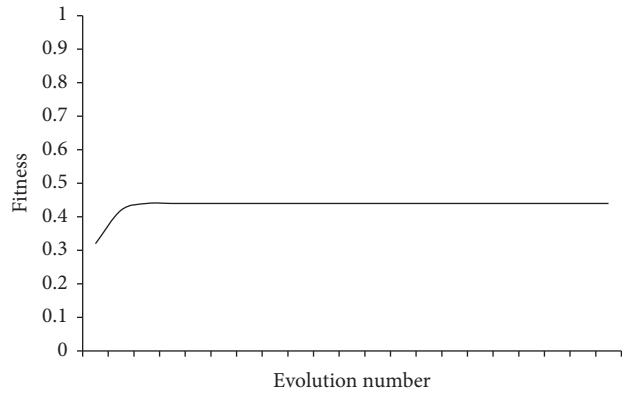


Figure 7: The algorithm cannot converge.

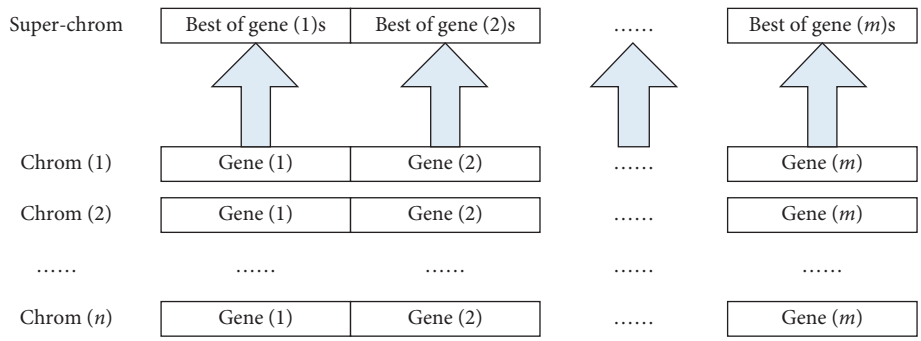


Figure 8: Composition of superchromosome.

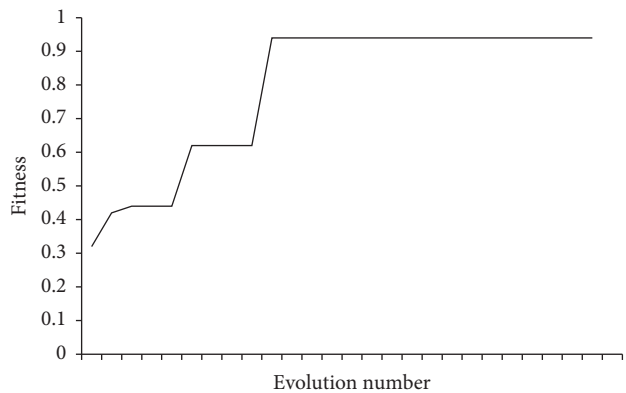


Figure 9: Superchromosome makes the fitness rise rapidly.

Table 4: Comparison of results of 4 groups in experiment 3.

Parameter	Exp1Value	Exp2Value	Exp3Value	Exp4Value
Logic gate	And, or, not	And, or, not	And, or, not	And, or, not
Values provided	Logic values	Logic values Voltage values	Logic values Current values	Logic values Voltage values Current values
Gene fitness function	$F = F_1$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_2$	$F = 0.8 \cdot F_1 + 0.2 \cdot F_3$	$F = 0.6 \cdot F_1 + 0.2 \cdot F_2 + 0.2 \cdot F_3$
Chromosome fitness	$F_{chrom} = (F_{gene1} + F_{gene2})/2$			
Evolution number	100000			
Max chrom-fitness	0.72	0.76	0.94	0.87
Max evolution no. when fitness stopped advance	324	276	809	1365

(1) After using superindividuals, the evolution can reach a very high level in a very short time, and then the speed of evolution will be significantly reduced.

(2) Using only a single feature or feature with direct correlation, the algorithm is difficult to obtain satisfactory fitness, and the reason has been analyzed in the results of experimental 1.

(3) The use of multiple features that lack direct correlation between each other helps to achieve higher fitness.

6. Conclusion

GEP algorithm based on the mixed features is proposed in this paper, when multiple features with no direct correlation between each other are used, even if some important parameters are missing in the test case, and the abnormal structure in the circuit can be found. For multioutput circuits, if it only simply decomposed into multiple single-output circuits to evolve separately, the algorithm will fall into a complete random search and cannot converge. In this paper, the concept of superindividual is proposed to solve this problem, so that the algorithm can converge smoothly in the circuit structure facing multi-input.

Data Availability

The data used to support the findings of this study can be obtained from https://pan.baidu.com/s/1z29JUVHv8Qx4-uasESnKMw (pwd: 55vd).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the University Research Start-Up Funds of Chengdu University of Information Technology (KYTZ201720), the Open Project of Center for Information in Biomedicine of School of Life Sciences and Technology, University of Electronic Science and Technology of China (SYFD061902K), and Sichuan Science and Technology Program (2019YFG0196).

References

[1] F. Courbon, P. Loubet-Moundi, J. J. A. Fournier, and A. Tria, "SEMBA, a SEM based acquisition technique for fast invasive hardware Trojan detection," in *Proceedings of the 2015 European Conference on Circuit Theory and Design (ECCTD)*, pp. 1–4, Trondheim, Norway, August 2015.

[2] C. X. Bao, D. Forte, and A. Srivastava, "On the application of one-class SVM to reverse engineering-based hardware Trojan detection," in *Proceedings of the 15th International Symposium on Quality Electronic Design (ISQED)*, pp. 47–54, Santa Clara, CA, USA, March 2014.

[3] C. X. Bao, D. Forte, and A. Srivastava, "On reverse engineering-based hardware Trojan detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, 2015.

[4] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Proceedings of the 2007 IEEE Symposium on Security and Piracy (SP'07)*, pp. 296–310, Berkeley, CA, USA, May 2007.

[5] K. Xiao, X. Zhang, and M. Tehranipoor, "A clock sweeping technique for detecting hardware Trojans impacting circuits delay," *IEEE Design & Test*, vol. 30, no. 2, pp. 26–34, 2013.

[6] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting Trojans through leakage current analysis using multiple supply pad I_{DDQ}s," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 893–904, 2010.

[7] A. N. Nowroz, K. Hu, F. Koushanfar, and S. Reda, "Novel techniques for high-sensitivity hardware Trojan detection using thermal and power maps," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1792–1805, 2014.

[8] B. Y. Zhou, R. Adato, M. Zangeneh et al., "Detecting hardware Trojans using backside optical imaging of embedded watermarks," in *Proceedings of the 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC'15)*, pp. 1–6, San Francisco, CA, USA, June 2015.

[9] J. He, Y. Zhao, X. Guo, and Y. Jin, "Hardware Trojan detection through chip-free electromagnetic side-channel statistical analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 10, pp. 2939–2948, 2017.

[10] D. Li, C. Shen, X. Dai et al., "Research on data fusion of adaptive weighted multi-source sensor," *Computers, Materials & Continua*, vol. 61, no. 3, pp. 1217–1231, 2019.

[11] Y. Jiang, W. Jiang, J. Chen et al., "A new method based on evolutionary algorithm for symbolic network weak unbalance," *Journal on Internet of Things*, vol. 1, no. 2, pp. 41–53, 2019.

[12] A. Y. Hamed, M. H. Alkinani, and M. R. Hassan, "A genetic algorithm optimization for multi-objective multicast routing," *Intelligent Automation & Soft Computing*, vol. 26, no. 6, pp. 1201–1216, 2020.

[13] F.-I. Chou, W.-H. Ho, and C.-H. Chen, "Niche genetic algorithm for solving multiplicity problems in genetic association studies," *Intelligent Automation & Soft Computing*, vol. 26, no. 3, pp. 501–512, 2020.

[14] H. Zhi and S. Liua, "A hybrid GABC-GA algorithm for mechanical design optimization problems," *Intelligent Automation and Soft Computing*, vol. 25, no. 4, pp. 815–825, 2019.

[15] C. Ferreira, "Gene expression programming: a new adaptive algorithm for solving problems," *Complex Systems*, vol. 13, no. 2, pp. 87–129, 2001.

[16] T. Higuchi, M. Murakawa, M. Iwata et al., "Evolvable hardware at function level," in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, pp. 187–192, Indianapolis, IN, USA, April 1997.

[17] T. Higuchi, M. Iwata, I. Kajitani et al., "Evolvable hardware and its application to pattern recognition and fault-tolerant systems," in *Towards Evolvable Hardware*, pp. 118–135, Springer, Berlin, Germany, 1996.

[18] V. Vassilev, D. Job, and J. Miller, "Towards the automatic design of more efficient digital circuits," in *Proceedings of the 2nd NASA/DOD Workshop on Evolvable Hardware*, pp. 151–160, Palo Alto, CA, USA, July 2000.

[19] G. W. Timothy and J. B. Peter, "Towards development in evolvable hardware," in *Proceedings of the 3rd NASA/DOD Workshop on Evolvable Hardware Pasadena*, pp. 241–250, Alexandria, VA, USA, July 2002.

[20] M. Erbo, R. Rossi, V. Liberali, and A. G. B. Tettamanzi, "Digital filter design through simulated evolution," in *Proceedings of the European Conference on Circuit Theory and Design*, pp. 389–393, Espoo, Finland, August 2001.

[21] H. Hemmi, J. Mizoguchi, and K. Shimohara, "Development and evolution of hard ware behaviors," in *Proceedings of the Artificial Life IV*, pp. 250–265, Cambridge, MA, USA, July 1994.

[22] B. Hounsell and T. Arslan, "A novel evolvable hardware framework for the evolution of high performance digital circuits," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00)*, pp. 525–529, Las Vegas, NV, USA, July 2000.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5516253, 7 pages
https://doi.org/10.1155/2021/5516253

Research Article

An Unsupervised Learning Method for the Detection of Genetically Modified Crops Based on Terahertz Spectral Data Analysis

Shubao Pan,[1] Binyi Qin,[2,3] Lvqing Bi,[3] Jincun Zheng,[3] Ruizhao Yang,[3] Xiaofeng Yang,[3] Yun Li,[4] and Zhi Li[1]

[1]*School of Electronic Engineering and Automation, Guilin University of Electronic Technology, Guilin 541004, China*
[2]*Key Laboratory of Complex System Optimization and Big Data Processing, Guangxi Colleges and Universities, Yulin Normal University, Yulin 537000, China*
[3]*School of Physics and Telecommunication Engineering, Yulin Normal University, Yulin 537000, China*
[4]*College of Chemistry and Food Science, Yulin Normal University, Yulin, Guangxi 537000, China*

Correspondence should be addressed to Binyi Qin; qby207@163.com and Yun Li; 675030346@qq.com

Received 10 January 2021; Revised 3 February 2021; Accepted 2 March 2021; Published 16 March 2021

Academic Editor: Liguo Zhang

Genetically modified crops have been planted commercially on a large scale since 1996. However, the food safety issue of genetically modified crops remains controversial. Conventional genetically modified crops' detection methods require a plenty of detective time and complex operations that cannot rapidly identify. Previous reports show that combining terahertz time-domain spectroscopy and supervised learning has advanced to identify genetically modified crops, but supervised learning requires large data to train the model. To solve the above problem, we proposed an unsupervised learning method, PCA-mean shift, to identify genetically modified crops. Principal component analysis was employed to reduce the absorbance data dimensionality. After principal component analysis, the first three principal components were used as the input of mean shift. At last, our proposed method had 100% identification accuracy, and *K*-means had 98.75% identification accuracy. The comparison results demonstrated that PCA-mean shift outperforms *K*-means. Therefore, PCA-mean shift combined with terahertz time-domain spectroscopy is a potential identification tool for genetically modified crops' identification.

1. Introduction

A genetically modified crop (GM crop) is a crop whose DNA has been modified using genetic engineering techniques [1]. Genetically modified technique has some advantages in resisting to virus and pests and herbicides [2, 3]. Since 1996, GM crops have been planted commercially on a large scale. There are 26 countries planting GM crops. International Service for the Acquisition of Agri-biotech Applications (ISAAA) reported that the total GM crops' planted area in the world is approximately 190 million hectares in 2018. Soybean, maize, cotton, and rape are the first four planted area GM crops in the world. However, the food safety issue of GM crops remains controversial [4–6]. Polymerase chain reaction [7], southern blot [8], western blot [9], and enzyme-linked immune sorbent assay [10] are conventional methods to identify GM crops, but these methods require plenty of detective time and complex operations. Thus, development of a practicable and effective analysis method for rapidly identifying GM crops is quite necessary.

Terahertz time-domain spectroscopy (THz-TDS) is a powerful detecting technique that operates in the frequency band from about 0.1 to 10 THz. It has been applied in the detection of biological and chemical molecules such as protein, amino acids, DNA, and harmful residues in agricultural and food products such as melamine, aflatoxin, pesticides, and antibiotics [11–19]. In recent years, some researchers reported the methods of identifying GM crops by using THz-TDS. Liu and Li [20] detected different GM cotton THz spectra by using the support vector machine (SVM). Xu et al. [21] reported discriminate

analysis (DA) and principal component analysis (PCA) have excellent performance to discriminate GM rice from non-GM rice from its parent. Chen et al. [22] proposed combining THZ-TDS and chemometrics to identify GM and non-GM sugar successfully. Wei et al. [23] achieved 96.67% accuracy with discrimination of GM rice by combining the THz-TDS image and chemometrics. In our previous research, we proposed a method combining SVM and multipopulation genetic algorithm (MPGA) for identifying GM cotton seed with THz spectroscopy [24]. However, there are two problems in the above research studies. First, most research studies for identifying GM crops are based on combining THz-TDS with supervised learning. As we know, supervised learning requires large data to train the model. Fewer sample data are one of the common problems in practice. Second, the above research studies identify only one crop. How to identify different GM crops is a challenging question.

Mean shift is an unsupervised learning method, which calculates the number of clusters automatically. It does not need to know the previous knowledge about the number of clusters and does not constrain the clusters' shape. Mean shift has been applied in target detection, target tracking, and image segmentation [25–28]. In recent years, Xing et al. [29] developed a colour clustering method for Chinese traditional costume image by mean shift. Wang et al. [30] discovered common visual patterns from two images by using mean shift to group together the close transformations in the space. Ai and Xiong [31] reported that it is able to enhance activation detections by incorporating mean shift and the temporal characteristics of fMRI. As far as we know, there are no studies using mean shift in THz-TDS.

In this paper, we proposed an unsupervised learning method PCA-mean shift to identify GM crops. THz spectrum is high-dimensional data. Firstly, we use PCA to reduce the THz spectrum dimension. And then, the first three principal components are chosen as the input of mean shift. Mean shift is an unsupervised learning method. At last, we compare our proposed method with K-means. The results indicate that PCA-mean shift is better than K-means, and PCA-mean shift is a potential method to identify GM crops.

2. Materials and Methods

2.1. Experimental System. The experimental system comprises a terahertz time-domain spectrometer Z-3 (Zomega Terahertz Corp., USA) and an ultrafast fiber laser (TOPTICA Photonics Inc., Germany). An ultrafast fiber laser generates 100 fs pulse widths with an 80 MHz repetition rate and 780 nm central wavelength. The spectral resolution is less than 5 GHz, and the peak dynamic range of the entire experimental system is better than 70 dB. The schematic diagram of the experimental system is shown in Figure 1. A laser beam is divided into two parts: a pump beam and a probe beam. The pump beam irradiates a photoconductive antenna to excite a terahertz beam. After that, the terahertz beam goes through the sample. And then, the terahertz beam meets the probe beam at electro-optic crystals ZnTe. The probe beam is modulated by the terahertz beam by the electro-optic effect. After transmitting through a quarter-wave plate (QWP) and a Wollaston prism (WP), the modulated probe beam is then detected by a set of balanced photodiodes (PD).

All measurements were carried out at room temperature (about 295 K) under the circumstance of a dry air-purged container with the relative humidity of less than 1%. Furthermore, we used the THz-TDS system in the transmission mode.

2.2. Sample Preparation. Two types of GM maize powder (GA21 and MIR604) were purchased from Shenzhen Excellence Biotechnology Co., Ltd. Non-GM maize powder was purchased from a local supermarket. Two types of GM cotton seeds (Lumianyan No.18 and Xinqiu No. k638) were obtained from Shandong Xinqiu Agricultural Science and Technology Co., Ltd.

In the two types of GM cotton seeds, the shell was removed, and they were crushed into powder, respectively. After that, five types of powder (GA21, MIR604, non-GM maize, Lumianyan No.18, and Xinqiu No. k638) were sieved by filtering laws using 100-eye sieves. The sieved powder was dried at 323 K for 1 hour and then was pressed into circular slices about 1.0 mm thick and 13 mm in diameter under a pressure of 8 MPa with a tablet press. At last, five types of specimens were obtained: GA21, MIR604, non-GM maize, Lumianyan No.18, and Xinqiu No. k638. For each type of specimen, 16 samples were prepared.

2.3. Principal Component Analysis. PCA is a common method for data dimension reduction. The fundamental idea of PCA is to approximate an original matrix X by a product of two small matrices shown in equation (1). X is an original data matrix consisting of n rows and p columns, U is a small matrix (called the score matrix) consisting of n rows and d columns, and L is another small matrix (called the loading matrix) consisting of p rows and d columns. T is the transpose of a matrix.

$$X = \mathrm{UL}^T. \tag{1}$$

The principal components (PCs) are determined based on the maximum variance criterion. Each subsequent PC describes a maximum of variance that is not modeled by the former components. According to this, most of the variance of the data is contained in the first PC. In the second component, there is more information than in the third one, and so on. Thus, a large fraction of the variance can be described with one, two, or three PCs, and the data can be visualized by plotting the PCs against each other. In our experiment, the original data X were THz spectra of samples.

2.4. Mean Shift. Mean shift is a density clustering algorithm without parameter estimation [32, 33]. It assumes that different clusters in a dataset accord with different probability density distributions. Mean shift can find the direction that the density of a sample point increases fastest. High density of the sample area corresponds to

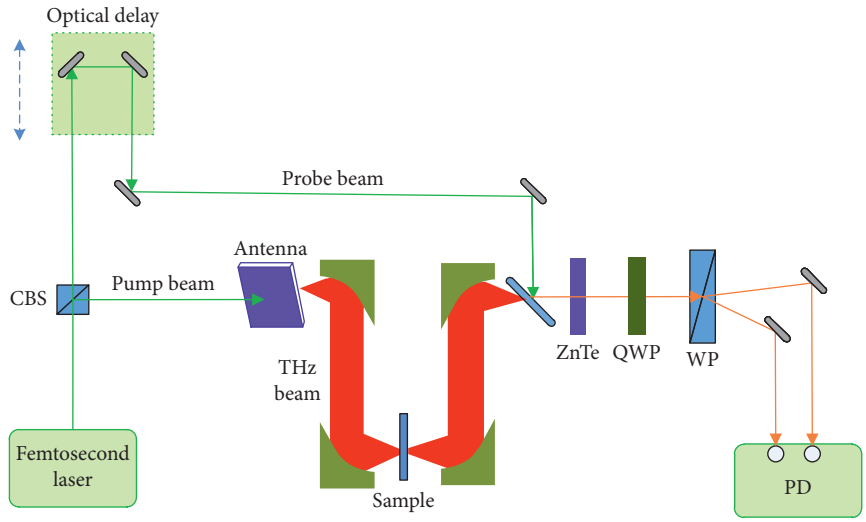


Figure 1: Schematic of the experimental apparatus.

the distribution of the maximum value of the sample points. These sample points will end up in a maximum density of local convergence. And the convergence to the same local maximum point is considered to be the same cluster member of the class.

Let x_i, $i = 1, 2, \ldots, n$, be a set of d-dimensional points in the space R^d. For a sample point x, the mean shift vector is defined as follows:

$$M_h(x) = \frac{1}{k} \sum_{x_i \in S_h} (x_i - x), \tag{2}$$

where k is the number of points that lie in S_h. S_h is a high-dimensional space with h radius. It is defined as

$$S_h(x) = \left\{ y \mid (y - x)^T (y - x) \leq h^2 \right\}. \tag{3}$$

The procedure of the mean shift algorithm is as follows:

Step 1: calculate the mean shift vector of each sample $m_h(x), m_h(x) = M_h(x) + x$

Step 2: move each sample with $m_h(x)$, such as $x_i = x_i + m_h(x_i)$

Step 3: repeat Step 1 until the sample point converges, $(m_h(x) = 0)$

Step 4: samples that converge to the same point are considered to be members of the same cluster

3. Results and Discussion

3.1. Spectroscopic Analysis. There are five types of specimens: GA21, MIR604, non-GM maize, Lumianyan No.18, and Xinqiu No. k638. For each type of the specimen, we measured 16 samples. Figure 2(a) displays the time-domain waveforms of five different specimens. Due to absorption and the refractive index difference between five specimens, the pulse amplitude and time delay are different. To compare the five types of specimens' time-domain waveforms, a novel type of plot-colour contour mapping of time-domain waveforms in terms of time is employed, as shown in Figure 2(b). Yellow means the pulse peak, and blue means the pulse valley. In Figure 2(b), the location of the pulse peak and valley between GA21 and non-GM maize is similar. Thus, it is impossible to immediately identify GA21 and non-GM maize by THz time-domain waveforms.

To obtain the absorbance, we translate the time-domain waveform of the sample and reference (air) into the frequency domain and then calculate it as follows [15]:

$$\text{absorbance}(\omega) = -\lg \left| \frac{E_s(\omega)}{E_{\text{ref}}(\omega)} \right|^2, \tag{4}$$

where ω is the frequency. And $E_s(\omega)$ and $E_{\text{ref}}(\omega)$ are the amplitude of the sample and reference signal in the frequency domain, respectively.

Figure 3(a) displays the terahertz absorbance spectra of five types of specimens in the frequency range of 0.4 THz to 1.4 THz. Figure 3(b) is the plot-colour contour mapping of absorbance spectra in terms of the frequency. Yellow represents the absorbance is strong, and blue represents the absorbance is weak. In Figure 3(b), the absorbance of MIR604 has strong absorption in the range of 1 THz to 1.4 THz. Lumianyan No.18 and Xinqiu No. k638 both have an absorption peak nearby 1.2 THz and 1.3 THz. For GA21 and non-GM maize, there is no sharp absorption peak in the range of 0.4 THz–1.4 THz. Comparing Figures 2(b) and 3(b), we find that absorbance spectra have more discrimination than time-domain waveforms. So, we selected absorbance spectra for further identification study.

3.2. Detection Results. In order to evaluate the performance of PCA-mean shift, confusion matrix and average accuracy were employed:

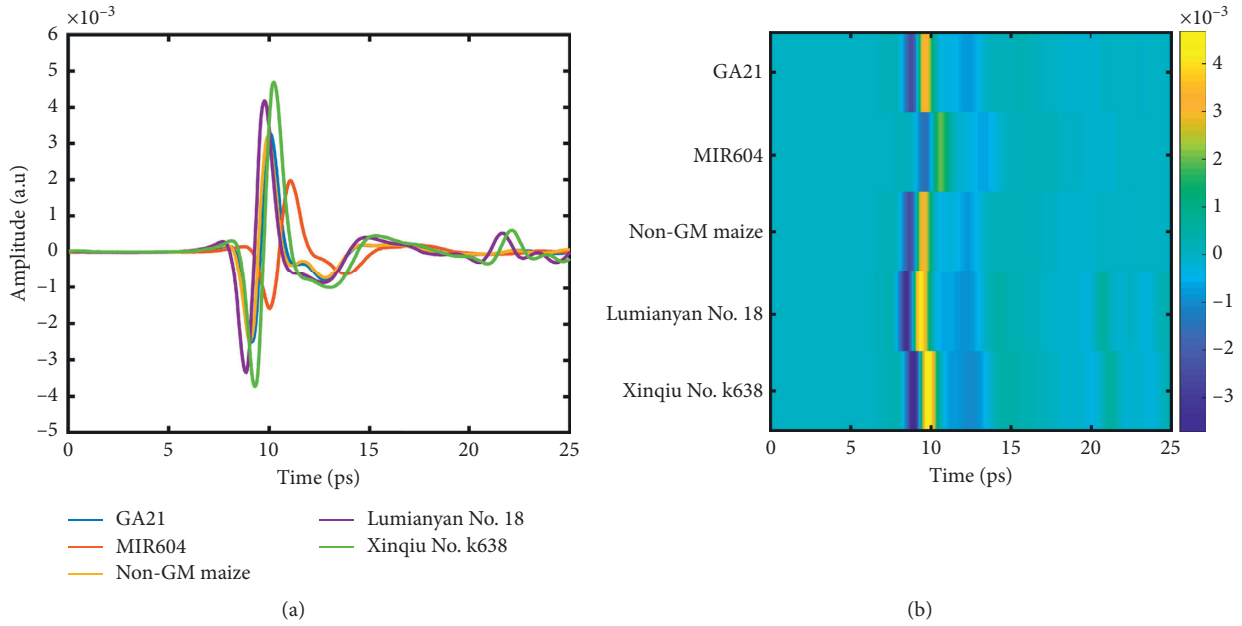


Figure 2: Time-domain waveforms of five types of specimens. (a) Time-domain waveforms. (b) Plot-colour contour mapping of time-domain waveforms.

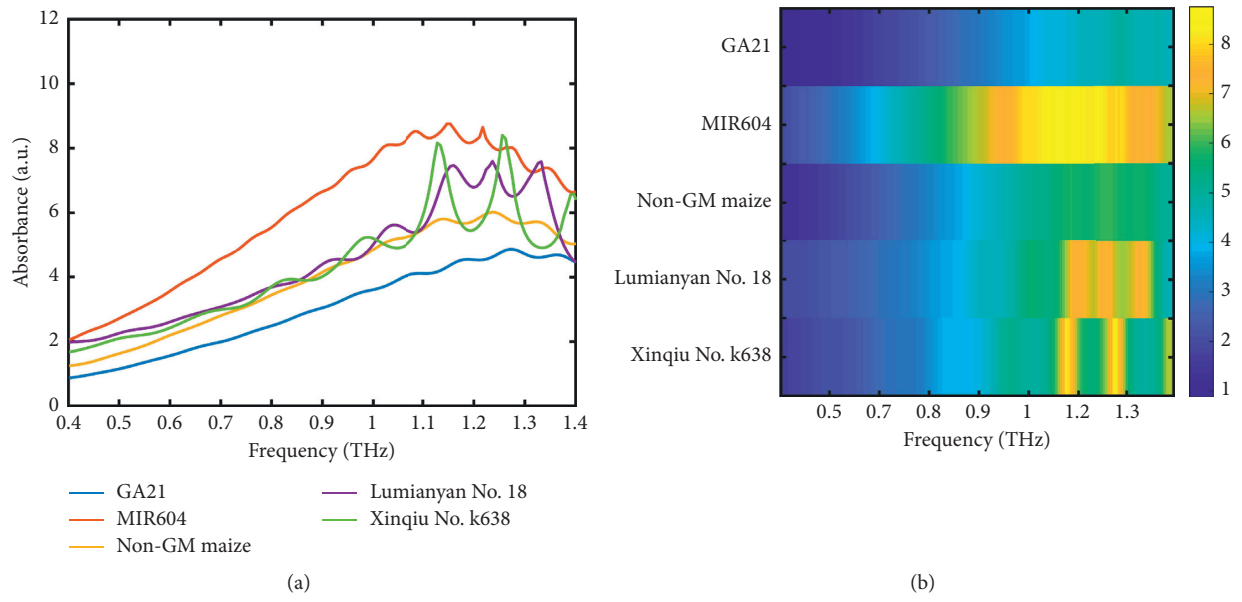


Figure 3: Absorbance of five types of specimens. (a) Absorbance. (b) Plot-colour contour mapping of five types of specimens' absorbance.

$$\text{average accuracy} = \frac{\text{the number of correct detection samples}}{\text{total of samples}}. \tag{5}$$

In addition, PCA-mean shift was compared with the common unsupervised learning method *K*-means. Firstly, we construct a dataset called Dataset1 by using absorbance spectra of five different specimens. Details of Dataset1 are shown in Table 1. And then, we use PCA to reduce the dimension of Dataset1.

By using PCA, absorption spectra were reduced from 80 dimensions to 3 dimensions. The variance contribution rate and cumulative variance contribution rate are listed in Table 2. Usually, as the cumulative variance contribution rate is large enough (typically $\geq 85\%$), the original dataset can be replaced approximately [34]. The cumulative variance contribution rate of the first three PCs is 90.43%. It means that the first three PCs contain major information of the original absorption spectra. Figure 4(a) shows the two-dimensional score of the first two PCs. After performing PCA, all the cotton seed samples are

Table 1: Composition of Dataset1.

Specimens	Number of samples	Region of frequency (THz)
GA21	16	0.4–1.4
MIR604	16	0.4–1.4
Non-GM maize	16	0.4–1.4
Lumianyan No. 18	16	0.4–1.4
Xinqiu No. k638	16	0.4–1.4

Table 2: The variance contribution rate and cumulative variance contribution rate.

Component	Variance (%)	Cumulative rate (%)
PC1	61.14	61.14
PC2	25.41	86.55
PC3	3.88	90.43

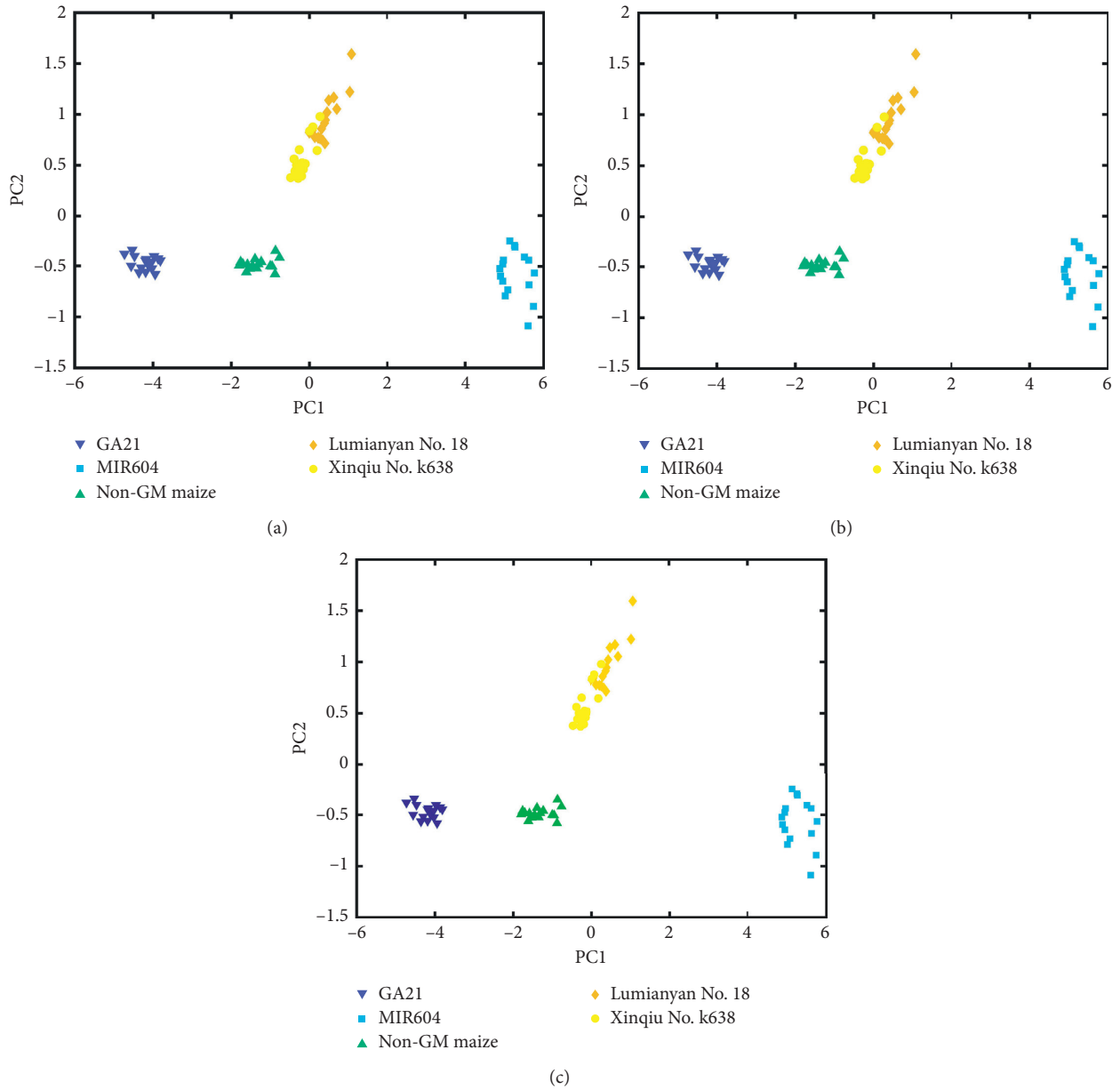


Figure 4: Results of detection. (a) PCA. (b) *K*-means. (c) PCA-mean shift.

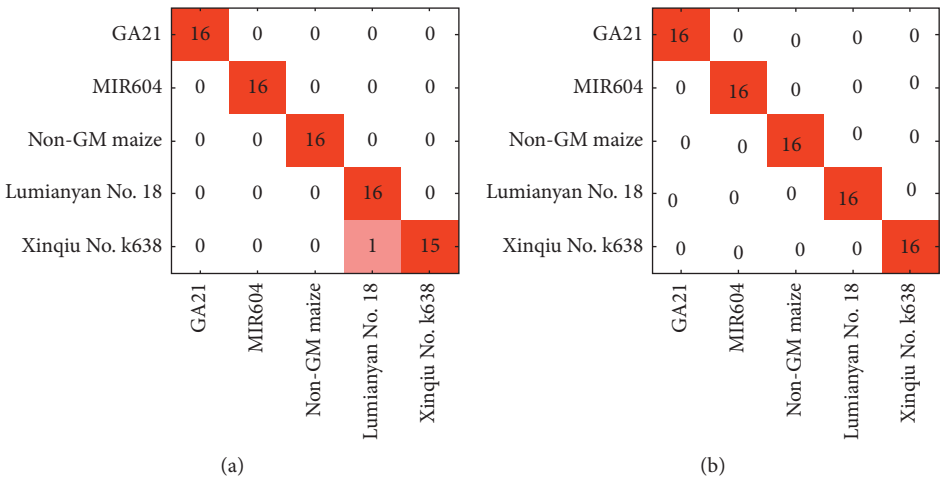


Figure 5: Confusion matrix. (a) *K*-means. (b) PCA-mean shift.

located on the upper half-plane, and all maize samples are located on the lower half-plane. It is easy to identify cotton and maize by their location. In the lower half-plane, GA21 is distributed in the left, non-GM maize is distributed in the middle, and MIR604 is distributed in the right. It is consistent with the absorption strength that GA21 has weak absorbance and MIR604 has strong absorbance. These three types of maize can be classified successfully. In Figure 4(a), these two types of cotton specimens are partly overlapped. It is due to that the absorbance of these two cotton specimens is similar. Thus, PCA is unable to identify Lumianyan No. 18 and Xinqiu No. k638 correctly.

After PCA, we used the first three PCs as the input of *K*-means and mean shift. In Figures 4(b) and 4(c), cotton and maize samples are divided into upper and lower half-planes. Thus, both methods are able to distinguish between cotton and maize. Because GA21, MIR604, and non-GM maize located on the lower half-plane have a large gap between each other, both *K*-means and PCA-mean shift can differentiate between the three types of maize. Unlike the sample points' distribution of the three types of maize, two types of cotton sample points overlap each other. The performance of classifying two types of cotton between *K*-means and PCA-mean shift is different. We employed the confusion matrix to evaluate the performance of these two methods, as shown in Figure 5. Figure 5(a) displays that one Xinqiu No. k638 sample is recognized as Lumianyan No. 18 by using *K*-means. Figure 5(b) shows that all the samples can be identified correctly by using PCA-mean shift. From the confusion matrix, the average accuracy of *K*-means and PCA-mean shift is 98.75% and 100%.

4. Conclusions

In this paper, an unsupervised learning method, PCA-mean shift, was proposed to identify two types of cotton and three types of maize with absorbance spectra in THz frequency. PCA was used to reduce the dimensionality of THz absorbance spectra. To compare our proposed method with *K*-means, confusion matrix and average accuracy were employed. The results indicated that PCA-mean shift had a higher average accuracy than *K*-means. Thus, PCA-mean shift combined with THz-TDS is a potential identification tool for GM crops' identification.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (Grant no. 62041111), Guangxi Natural Science Foundation (Grant no. 2019GXNSFBA245076), Guangxi Key Laboratory of Automatic Detecting Technology and Instruments Foundation (Grant nos. YQ19208 and YQ20207), Opening Foundation of Yulin Research Institute of Big Data (Grant no. 2020YJKY04), Major Cooperative Project between Yulin Municipal Government and Yulin Normal University (Grant no. YLSXZD2019015), Doctoral Scientific Research Foundation of Yulin Normal University (Grant no. G2019K02), and Yulin Normal University Research Grant (Grant no. 2015YJYB06).

References

[1] A. Repellin, M. Baga, P. Jauhar, and R. Chibbar, "Genetic enrichment of cereal crops via alien gene transfer: new challenges," *Plant Cell Tissue and Organ Culture*, vol. 64, no. 2-3, pp. 159–183, 2001.

[2] X. K. Morin, "Genetically modified food from crops: progress, pawns, and possibilities," *Analytical and Bioanalytical Chemistry*, vol. 392, no. 3, pp. 333–340, 2008.

[3] H. Azadi, M. Ghanian, O. M. Ghoochani et al., "Genetically modified crops: towards agricultural growth, agricultural

development, or agricultural sustainability?" *Food Reviews International*, vol. 31, no. 3, pp. 195–221, 2015.

[4] A. Bakshi, "Potential adverse health effects of genetically modified crops," *Journal of Toxicology and Environmental Health, Part B*, vol. 6, no. 3, pp. 211–225, 2003.

[5] G. Peterson, S. Cunningham, L. Deutsch et al., "The risks and benefits of genetically modified crops: a multidisciplinary perspective," *Conservation Ecology*, vol. 4, no. 1, pp. 1–4, 2000.

[6] M. Kramkowska, T. Grzelak, and K. Czyzewska, "Benefits and risks associated with genetically modified food products," *Annals of Agricultural and Environmental Medicine*, vol. 20, no. 3, pp. 413–419, 2013.

[7] H. Cheng, W. Jin, H. Wu et al., "Isolation and PCR detection of foreign DNA sequences in bee honey raised on genetically modified Bt (Cry1Ac) cotton," *Food and Bioproducts Processing*, vol. 85, no. C2, pp. 141–145, 2007.

[8] M. S. McCabe, J. B. Power, A. M. M. de Laat, and M. R. Davey, "Detection of single-copy genes in DNA from transgenic plants by nonradioactive southern blot analysis," *Molecular Biotechnology*, vol. 7, no. 1, pp. 79–84, 1997.

[9] X. Xiao, H. Wu, X. Zhou et al., "The combination of quantitative PCR and western blot detecting CP4-EPSPS component in roundup ready soy plant tissues and commercial soy-related foodstuffs," *Journal of Food Science*, vol. 77, no. 6, pp. 603–608, 2012.

[10] G. Liu, W. Su, Q. Xu, M. Long, J. Zhou, and S. Song, "Liquid-phase hybridization based PCR-ELISA for detection of genetically modified organisms in food," *Food Control*, vol. 15, no. 4, pp. 303–306, 2004.

[11] M. D. King, P. M. Hakey, and T. M. Korter, "Discrimination of chiral solids: a terahertz spectroscopic investigation ofl- anddl-serine," *The Journal of Physical Chemistry A*, vol. 114, no. 8, pp. 2945–2953, 2010.

[12] Y. Ueno, K. Ajito, N. Kukutsu, and E. Tamechika, "Quantitative analysis of amino acids in dietary supplements using terahertz time-domain spectroscopy," *Analytical Sciences*, vol. 27, no. 4, p. 351, 2011.

[13] I. Maeng, S. H. Baek, H. Y. Kim, G.-S. Ok, S.-W. Choi, and H. S. Chun, "Feasibility of using terahertz spectroscopy to detect seven different pesticides in wheat flour," *Journal of Food Protection*, vol. 77, no. 12, pp. 2081–2087, 2014.

[14] J. El Haddad, B. Bousquet, L. Canioni, and P. Mounaix, "Review in terahertz spectral analysis," *TrAC Trends in Analytical Chemistry*, vol. 44, pp. 98–105, 2013.

[15] B. Qin, Z. Li, Z. Luo, H. Zhang, and Y. Li, "Feasibility of terahertz time-domain spectroscopy to detect carbendazim mixtures wrapped in paper," *Journal of Spectroscopy*, vol. 2017, no. 7, 8 pages, Article ID 6302868, 2017.

[16] B. Qin, Z. Li, F. Hu et al., "Highly sensitive detection of carbendazim by using terahertz time-domain spectroscopy combined with metamaterial," *IEEE Transactions on Terahertz Science and Technology*, vol. 8, no. 2, pp. 149–154, 2018.

[17] B. Cao, H. Li, E. Cai, and M. Fan, "Determination of pesticides in flour by terahertz time-domain spectroscopy (thz-tds) with voigt function fitting and partial least squares (pls) analysis," *Analytical Letters*, vol. 54, pp. 1–12, 2020.

[18] B. Cao, H. Li, M. Fan, W. Wang, and M. Wang, "Determination of pesticides in a flour substrate by chemometric methods using terahertz spectroscopy," *Analytical Methods*, vol. 10, no. 42, pp. 5097–5104, 2018.

[19] W. Liu, P. Zhao, C. Wu, C. Liu, J. Yang, and L. Zheng, "Rapid determination of aflatoxin B1 concentration in soybean oil using terahertz spectroscopy with chemometric methods," *Food Chemistry*, vol. 293, pp. 213–219, 2019.

[20] J. Liu and Z. Li, "The terahertz spectrum detection of transgenic food," *Optik*, vol. 125, no. 23, pp. 6867–6869, 2014.

[21] W. Xu, L. Xie, Z. Ye et al., "Discrimination of transgenic rice containing the Cry1Ab protein using terahertz spectroscopy and chemometrics," *Scientific Reports*, vol. 5, p. 1115, 2015.

[22] T. Chen, Z. Li, X. Yin, F. Hu, and C. Hu, "Discrimination of genetically modified sugar beets based on terahertz spectroscopy," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 153, pp. 586–590, 2016.

[23] W. Liu, C. Liu, X. Hu, J. Yang, and L. Zheng, "Application of terahertz spectroscopy imaging for discrimination of transgenic rice seeds with chemometrics," *Food Chemistry*, vol. 210, pp. 415–421, 2016.

[24] B. Qin, Z. Li, T. Chen, and Y. Chen, "Identification of genetically modified cotton seeds by terahertz spectroscopy with mpga-svm," *Optik*, vol. 142, pp. 576–582, 2017.

[25] J. Michel, D. Youssefi, and M. Grizonnet, "Stable mean-shift algorithm and its application to the segmentation of arbitrarily large remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 2, pp. 952–964, 2014.

[26] J. Kim, S. Lee, G. Lee, Y. Park, and Y. Hong, "Using a method based on a modified k-means clustering and mean shift segmentation to reduce file sizes and detect brain tumors from magnetic resonance (mri) images," *Wireless Personal Communications*, vol. 89, no. 3, pp. 993–1008, 2016.

[27] Q. Wei, T. Dai, T. Ma, Y. Liu, and Y. Gu, "Crystal identification in dual-layer-offset doi-pet detectors using stratified peak tracking based on svd and mean-shift algorithm," *IEEE Transactions on Nuclear Science*, vol. 63, no. 5, pp. 2502–2508, 2016.

[28] H. Cho, S. J. Kang, and Y. H. Kim, "Image segmentation using linked mean-shift vectors and global/local attributes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 10, p. 1, 2017.

[29] L. Xing, J. Zhang, H. Liang, and Z. Li, "Intelligent recognition of dominant colors for Chinese traditional costumes based on a mean shift clustering method," *The Journal of the Textile Institute*, vol. 109, no. 10, pp. 1304–1314, 2018.

[30] L. Wang, D. Tang, Y. Guo, and M. N. Do, "Common visual pattern discovery via nonlinear mean shift clustering," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5442–5454, 2015.

[31] L. Ai and J. Xiong, "Temporal-spatial mean-shift clustering analysis to improve functional MRI activation detection," *Magnetic Resonance Imaging*, vol. 34, no. 9, pp. 1283–1291, 2016.

[32] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[33] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[34] T. Chen, Z. Li, and W. Mo, "Identification of biomolecules by terahertz spectroscopy and fuzzy pattern recognition," *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, vol. 106, pp. 48–53, 2013.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6699974, 14 pages
https://doi.org/10.1155/2021/6699974

Research Article

Distributed Functional Signature with Function Privacy and Its Application

Muhua Liu, Lin Wang, Qingtao Wu, and Jianqiang Song

Control Science and Engineering Postdoctoral Mobile Station, Henan University of Science and Technology, Luoyang, China

Correspondence should be addressed to Qingtao Wu; wqt8921@haust.edu.cn

Received 27 December 2020; Revised 23 February 2021; Accepted 27 February 2021; Published 12 March 2021

Academic Editor: Liguo Zhang

We introduce a novel notion of distributed functional signature. In such a signature scheme, the signing key for function f will be split into n shares sk_f^i and distributed to different parties. Given a message m and a share sk_f^i, one can compute locally and obtain a pair signature $(f_i(m), \sigma_i)$. When given all of the signature pairs, everyone can recover the actual value $f(m)$ and corresponding signature σ. When the number signature pairs are not enough, nobody can recover the signature $(f(m), \sigma)$. We formalize the notion of function privacy in this new model which is not possible for the standard functional signature and give a construction from standard functional signature and function secret sharing based on one-way function and learning with error assumption. We then consider the problem of hosting services in multiple untrusted clouds, in which the verifiability and program privacy are considered. The verifiability requires that the returned results from the cloud can be checked. The program privacy requires that the evaluation procedure does not reveal the program for the untrusted cloud. We give a verifiable distributed secure cloud service scheme from distributed functional signature and prove the securities which include untrusted cloud security (program privacy and verifiability) and untrusted client security.

1. Introduction

Digital signature schemes were introduced by Diffie and Hellman [1]. In a digital signature scheme, it has a secret signing key and a corresponding verification key. Only the person who has the secret key can sign a message, and any person can verify the signature with the verifiable key. Goldwasser et al. [2] gave a standard secure definition. A signature scheme is unforgeable against chosen message attack if an adversary produces a valid signature of any message with at most negligible probability in probabilistic polynomial time. The adversary is allowed to query signatures for a polynomial number of messages of his choice. Subsequently, it appeared to have many other forms of signatures, such as blind signature [3], group signature [4], ring signature [5], identity-based signature [6], homomorphic signature [7, 8], and so on.

Functional signatures (FSs) were proposed by Boyle et al. [9]. In a functional signature scheme, in addition to a master key which can sign any messages belonging to the message space, it also has a signing key sk_f for any function f, which are derived from the master key. Using the signing key sk_f, one can sign any message in the range of f. The security requires that the probability of producing a valid signature on message m does not exceed a negligible function for any probability polynomial adversary which can query the signing key for functions $f_1, f_2, \ldots, f_\ell$ of his choice and signatures of messages $m_1, m_2, \ldots, m_n$ of his choice. A valid signature on m means that m does not equal to one of the queried messages $m_1, m_2, \ldots, m_n$ or m does not belong to range of one of the queried functions $f_1, f_2, \ldots, f_\ell$. For a functional signature scheme, a desirable property is function privacy which requires a signature should not reveal the function. A typical application is the signed photo-processing software [9]; the owner of photos want to remove red eyes or color scale, but does not allow more significant changes. At the same, the owner wishes the signed photos do not reveal the original image.

Boyle et al. [9] firstly gave a construction based on the existence of one-way functions, which does not satisfy the property of function privacy. Then, they transformed it into a construction that satisfied the function privacy by the noninteractive zero-knowledge arguments of knowledge for NP languages which is based on nonfalsifiable assumptions. In their secure definition of function privacy, it requires that the distribution of signatures on a message m generated via different keys sk_f to be computationally indistinguishable. While this model restricted the application of the scheme, it is not clear how could this weakened notion be applied in the general setting. For example, a software company develops software and delegates the computation ability to a server. The company wishes it does not reveal the core function during the computational procedure and the returned results can be verified. In a functional signature scheme, standard function privacy cannot possibly satisfy this scenario. Since the attacker who has a signing key sk_f can generate the signature $(\sigma, f(m))$ for the message m on the fly, the attacker can obtain the evaluations of $f(m_1), \ldots, f(m_n)$ for the messages $m_1, \ldots, m_n$ of its choices. It will reveal some functions of the software by the evaluation pairs $\{(m_i, f(m_i))\}_{i=1}^{n}$.

Therefore, we try to find a realistic model that allows us to approach function privacy for functional signature based on some standard assumption, such as the existing one-way function and learning with error assumption.

1.1. Our Contributions. In this paper, we introduce the definition of distributed functional signature scheme. In such a model, the signing key for a function f will be split into n shares sk_f^i and distributed to different parties. Given a secret signing key sk_f^i, anyone can compute locally and obtain a pair signature $(f_i(m), \sigma_i)$. When given n signatures $\{(f_i(m), \sigma_i)\}_{i=1}^{n}$ on the message m, everyone can reconstruct the function evaluation $f(m)$ and the corresponding signature σ. While given less than n signatures $\{(f_i(m), \sigma_i)\}_{i\in S}$ satisfying $|S| < n$, anyone cannot reconstruct the evaluation $f(m)$ and the corresponding signature σ. In our secure definition of function privacy, we require the attacker to not distinguish the distribution of signing key $\{\mathrm{sk}_f^i\}_{i\in S}$ generated via different function f, where $|S| < n$. The attacker is allowed to query key generation oracle and evaluation oracle. This new definition of distributed functional signature generalizes the notion of threshold signature to the setting of functional signature, enables the participants to recover an evaluation of the function f on input m, and verifies the result by the corresponding signature together.

In our new model, it provides a possibility of function privacy in the setting of functional signature. Specifically, when given less than n signing key sk_f^i for function f, the adversary can only know $f_i(m)$ corresponding to the share function f_i which is not enough to determine $f(m)$. We give a construction for distributed functional signature by transforming any functional signature which does not support function privacy into a distributed version which satisfies both unforgeability and function privacy via function secret sharing [10]. Our scheme can be constructed based on the one-way function and learning with error assumption.

We remark here that our notion of function privacy is different from the standard function privacy [9]. The latter mainly considers that single signature does not reveal the function, while we consider that the signature process does not reveal the function by splitting each function into secret key shares. In our scheme, the attacker only gets at most $n-1$ secret key shares and thus obtains at most $n-1$ signatures $\{(f_i(m), \sigma_i)\}_{i\in S}$ satisfying $|S| < n$. Anyone cannot reconstruct the value $f(m)$. Therefore, it satisfies the function privacy.

Hosting services securely in multiple untrusted clouds. In recent years, cloud computing has become a very hot research area in cryptography. Boneh et al. [11] first studied hosting service in the cloud. They considered the security which includes protecting program information and client's inputs against untrusted cloud and protecting program information and authorization procedure against untrusted clients. It has wide applications to protect program information against untrusted client and cloud, such as software protected. Fan and Tang [12] gave a construction from distributed functional encryption. However, it requires that the cloud is semihonest. Sometimes, the cloud may run a fast but incorrect computation due to a financial incentive. Therefore, we also consider another property of verifiability which requires that the returned results from the cloud can be checked in the untrusted cloud model.

In addition, the construction of Fan and Tang [12] is based on the generic functional encryption. As far as we know, most works have approached the problem of constructing generic functional encryption by proposing a candidate under the existing of indistinguishability obfuscation [13]. Meanwhile, the construction of Boneh et al. [11] is also based on the existing indistinguishability obfuscation. However, indistinguishability obfuscation is not a particularly appealing assumption since the security of constructions relies on an exponential number of assumptions [14]. Recently, numerous attacks on multilinear maps [15] have reduced the community's confidence in the security of existing construction of indistinguishability obfuscation. In this work, we provide a construction of secure cloud service scheme based on some more general assumptions, such as the existing one-way function and learning with error assumption.

A secure cloud service scheme satisfying the verifiability can be constructed by a distributed functional signature scheme and a standard signature scheme. The standard signature scheme is used to authenticate the client. In order to achieve the verifiability, the service first generates some signing key $\mathrm{sk}_{\widehat{P}}^i$ for the program $\widehat{P} = x|P(x)$. When the client receives the returned results from clouds, it can get the evaluation (y, σ) by the reconstruction algorithm of distributed signature scheme. To prove that $y = x|P(x)$, the client verifies the signature σ of y, where the signature σ could only be obtained if y is in the range of $\widehat{P}$.

1.2. Related Works. The concept of identity-based signature was proposed by Shamir in [16]. Bellare and Fuchsbauer [17] introduced policy-based signatures, where a signer can only sign messages satisfying some authority-specified policy. Backes et al. [18] introduced delegatable functional signatures which support the delegation of signing capabilities to another party with respect to a functionality. This new notion unifies several signature primitives, such as policy-based signature, functional signature, identity-based signature, and blind signature.

Homomorphic signature was first proposed by Johnson et al. [19], which was initially designed to establish authentication in network coding. In addition, it can also be used to authenticate the stored data. In a homomorphic signature scheme, a holder which has the signature pairs (m_0, σ_0) and (m_1, σ_1) can construct a new signature σ on the value $f(m_0, m_1)$ for some function f without the signing key. Freeman [20] gave a linearly homomorphic signature scheme which allows authentication of linear functions on signed data. Catalano et al. [21] constructed a homomorphic signature scheme for polynomial functions.

The concept of functional encryption comes from the work of Sahai and Waters [22]. The definition of simulation-based security was firstly proposed by Boneh et al. [23]. Garg et al. [13] firstly proposed a construction for functional encryption which supported all polynomial size circuits. Goldwasser et al. [24] constructed a succinct functional encryption scheme based on reusable garbled circuits and identity-based encryption for any polynomial-time function. Subsequently, Goldwasser et al. [25] introduced the notion of multi-input functional encryption, which supported the multi-input functions, and gave a construction based on indistinguishability obfuscation. In a multi-input functional encryption, the decryption key sk_f for a function f takes multiple ciphertexts as input and outputs a function evaluation. Besides, there are many other functional encryptions for inner-product functions [26–28].

Boneh et al. [11] gave a construction of hosting service in the cloud. Their construction relied on indistinguishability obfuscation $i\mathcal{O}$ and restricted the number of colluded clients for the security. Fan and Tang [12] presented a new definition of distributed public key functional encryption. In their scheme, the decryption key for a function f is split into several sharing key sk_i^f. Given a ciphertext that encrypts a message m and a sharing key sk_i^f, it can evaluate a shared value y_i which reveals nothing about the plaintext and the value of $f(m)$. One can recover the value of $f(m)$ by adding all the shared values y_i. With this approach, it can achieve function privacy which is not possible in the setting of regular public key functional encryption. Then, they considered the problem of hosting services in the untrusted cloud. Applying the function private distributed public key functional encryption to the setting of hosting service in multiple clouds, it can remove the restriction that the number of corrupted clients has to be bounded and known in the previous works.

Boyle et al. [9] firstly proposed functional signatures, which allowed an authority to generate signing keys corresponding to various functions such that a user with a signing key sk_f can sign the image of function f on a message m. Okamoto and Takashima [29] firstly introduced the concept of a decentralised multiauthority functional signatures, which supports nonmonotone access structures combined with inner-product relations. Liang and Mitrokotsa [30] generalised the definition of decentralised multiauthority functional signature for more general policy functions. Datta et al. [31] introduced the concept of functional signcryption, which provided the functionalities of both functional encryption and functional signature. Pan et al. [32] introduced the notion of hierarchical functional signcryption which expanded the scope of functional signcryption in hierarchical access-control application. Li et al. [33] introduced the notion of private functional signatures, in which the signing key sk_f can be used to generate a signature $\sigma_{f(x)}$ on the ciphertext c_x, where c_x is a ciphertext for message x.

Blockchains originate from Bitcoin and are essentially a decentralised database that uses peer-to-peer network. It is a new application mode of computer technology such as distributed data storage, point-to-point transmission, and consensus mechanism. This makes blockchains potentially suitable for recording events, medical records, and other management activities. Although it can realize the function of software protection, it does not possess computing capabilities. In this work, our goal is to achieve software protection while ensuring software availability.

Notation. In what follows, we will denote with $\lambda \in \mathcal{N}$ a security parameter. We say $\mathrm{negl}(\lambda)$ is negligible if $|\mathrm{negl}(\lambda)| < (1/\mathrm{poly}(\lambda))$ holds for all polynomials $\mathrm{poly}(\lambda)$ and all sufficiently large λ. Denote $[n]$ as the set $\{1, 2, \ldots, n\}$. We abbreviate probabilistic polynomial time as PPT. Denote $|S|$ as the number of elements in the set S.

2. Preliminaries

In this section, we present definition for various cryptographic primitives that we will use in our construction of distributed functional signature. We assume familiarity with standard signature satisfying unforgeability against adaptively chosen message attack. Below, we first recall the notions of function secret sharing and functional signature.

2.1. Function Secret Sharing. We give the formal definition following the syntax of [10]. An (n, n)–function secret sharing scheme for a function family $\mathcal{F}$ is a tuple of algorithms (FSS.Setup, FSS.ShareGen, FSS.Recon) described as follows:

(i) $\mathrm{FSS.Setup}(1^\lambda, n, \mathcal{F}) \longrightarrow$ FSS.pp: on inputting the security parameter λ, the number of sharers n, and the description of function family $\mathcal{F}$, this algorithm outputs the public parameters FSS.pp.

(ii) $\mathrm{FSS.ShareGen}(\mathrm{FSS.pp}, f) \longrightarrow \{f_i\}_{i=1}^n$: on inputting the parameters FSS.pp and a function $f \in \mathcal{F}$, this algorithm outputs n shares $\{f_i\}_{i=1}^n$ for function f.

(iii) FSS.Recon$(\text{FSS.pp}, \{f_i(x)\}_{i=1}^n) \longrightarrow f_i(x)$: on inputting n values $\{f_i(x)\}_{i=1}^n$ which evaluated each function share f_i on x, this algorithm reconstructs all the share values $\{f_i(x)\}_{i=1}^n$ and outputs $f(x)$.

Definition 1. Correctness: an (n, n)–function secret sharing scheme for the function family $\mathcal{F}$ is correct, if for any function $f \in \mathcal{F}, x \in \mathcal{D}_f$, FSS.pp $\longleftarrow$ FSS.Setup$(1^\lambda, n, \mathcal{F})$, and $\{f_i\}_{i=1}^n \longleftarrow$ FSS.ShareGen(FSS.pp, f), we have

$$\Pr[f(x) = \text{FSS.Recon}(\text{FSS.pp}, \{f_i(x)\}_{i=1}^n)] \geq 1 - \text{negl}(\lambda). \tag{1}$$

Definition 2. Security: we say an (n, n)–function secret sharing scheme is secure if for any PPT adversary $\mathcal{A}$, the advantage in the following game is negligible:

(i) The challenger generates FSS.PP $\longleftarrow$ FSS.Setup $(1^\lambda, n, \mathcal{F})$ and sends FSS.pp to the adversary $\mathcal{A}$.

(ii) The adversary outputs $(S, f^0, f^1) \longleftarrow \mathcal{A}(\text{FSS.pp}, \lambda)$, where $f^0, f^1 \in \mathcal{F}$, $\mathcal{D}_{f^0} = \mathcal{D}_{f^1}$, and $S \in [n], |S| = n - 1$.

(iii) The challenger randomly chooses a bit $b \longleftarrow \{0, 1\}$, computes $\{f_i^b\}_{i=1}^n \longleftarrow$FSS.ShareGen(FSS.pp, f^b), and sends $\{f_i^b\}_{i \in S}$ to the adversary $\mathcal{A}$.

(iv) $\mathcal{A}$ outputs a guess $b' \longleftarrow \mathcal{A}(\text{FSS.pp}, \{f_i^b\}_{i \in S})$. The advantage of $\mathcal{A}$ is defined as Adv $= (\Pr[b' = b] - 1/2)$.

2.2. Functional Signatures. We describe the definition of a functional signature scheme, which is proposed by Boyle et al. in [9]. A functional signature scheme for a message space $\mathcal{M}$ and a function family $\mathcal{F} = \{f: \mathcal{D}_f \longrightarrow \mathcal{M}\}$ consists of four algorithms (FS.Setup, FS.KeyGen, FS.Sign, FS.Very):

(i) FS.Setup$(1^\lambda) \longrightarrow$ (FS.mvk, FS.msk): on inputting the security parameter, the setup algorithm outputs the master verification key FS.mvk and the master signing key FS.msk.

(ii) FS.KeyGen(FS.msk, f) $\longrightarrow \text{sk}_f$: on inputting the master signing key FS.msk and a function $f \in \mathcal{F}$, the key generation algorithm outputs a constrained signing key sk_f, which just signs the message $f(m)$ for any $m \in \mathcal{D}_f$.

(iii) FS.Sign$(f, \text{sk}_f, m) \longrightarrow (f(m), \sigma)$: on inputting the function f, constrained signing key sk_f, and a message $m \in \mathcal{D}_f$, the signing algorithm outputs a pair signature $(f(m), \sigma)$.

(iv) FS.Very(FS.mvk, m^*, σ) $\longrightarrow \{0, 1\}$: on inputting the verification algorithm FS.mvk, a message m^*, and a signature σ, the verifiable algorithm outputs "1" or "0," where 1 indicates that the signature is valid.

Definition 3. Correctness: for any $f \in \mathcal{F}$ and $m \in \mathcal{D}_f$, (FS.msk, FS.mvk)$\longleftarrow$FS.Setup(1^λ), $\text{sk}_f \longleftarrow$FS.KeyGen (FS.msk, f), $(m^*, \sigma) \longleftarrow$FS.Sign$(f, \text{sk}_f, m)$, it holds that

$$\text{FS.Very}(\text{FS.mvk}, m^*, \sigma) = 1. \tag{2}$$

Definition 4. Unforgeability: the scheme is unforgeable if the successful probability of any PPT algorithm $\mathcal{A}$ in the following game is negligible:

(i) The challenger runs (FS.mvk, FS.msk)$\longleftarrow$ FS.Setup(1^λ) and sends FS.mvk to the adversary $\mathcal{A}$.

(ii) The adversary $\mathcal{A}$ can query a key generation oracle and a signing oracle. The challenger initializes a dictionary indexed by tuples $(f, j) \in \mathcal{F} \times \mathcal{N}$, which contains the signing keys: $\text{sk}_f \longleftarrow$FS. KeyGen (FS.msk, f).

(iii) Key generation oracle: on inputting (f, j), the challenger runs as follows:

(1) If there is an entry for (f, j) in the dictionary, then output the corresponding key sk_{fj}.

(2) Otherwise, sample a fresh key $\text{sk}_{fj} \longleftarrow$FS.KeyGen (FS.msk, f), update the dictionary $(f, j) \longrightarrow \text{sk}_{fj}$, and output sk_{fj}.

(iv) Signing oracle: on inputting (f, j, m), the challenger runs as follows:

(1) If there is an entry for the key (f, j) in the dictionary, then output the signature $\sigma \longleftarrow$FS.Sign(f, sk_{fj}, m).

(2) Otherwise, sample a fresh key $\text{sk}_{fj} \longleftarrow$FS.KeyGen(FS.msk, f), update the dictionary $(f, j) \longrightarrow \text{sk}_{fj}$, and output the signature $\sigma \longleftarrow$FS.Sign(f, sk_{fj}, m).

(v) The adversary succeeds if it outputs a signature (m^*, σ) such that

(1) FS.Very(FS.mvk, m^*, σ) = 1.

(2) There does not exist m such that $m^* = f(m)$ for any f which was sent as a query to the key generation oracle.

(3) There does not exist a (f, m) such that (f, m) was queried to the signing oracle and $m^* = f(m)$.

Lemma 1 (see [9]). *A functional signature scheme can be constructed based on any one-way function that supports signing keys for any function f which is computed by a polynomial-sized circuit. This scheme satisfies the property of unforgeability, but not function privacy.*

2.3. General Aggregate Signatures. A general aggregate signature scheme [34] consists of four algorithms (AS.KeyGen, AS.Sign, AS.Agg, AS.Very). The key generation algorithm and signing algorithm are the same as a standard digital

signature. In the aggregate algorithm, it produce a new signature by $\sigma \longleftarrow \mathrm{Agg}((\mathrm{pk}_1, m_1, \sigma_1), (\mathrm{pk}_n, m_n, \sigma_n))$. Given a sequence of public key, message, and signature triples, anyone can yield an aggregate signature σ. Through $\mathrm{Very}((\mathrm{pk}_1, m_1), \ldots, (\mathrm{pk}_n, m_n), \sigma)$, anyone can verify the correctness of generation aggregate signature.

In addition to satisfying unforgeability under chosen message attack of the standard signature, it needs to satisfy aggregate unforgeability. The security requires that it is computationally infeasible to produce an aggregate forgery for a PPT adversary which can corrupt at most $n-1$ players. The detailed description of security is given as follows.

Definition 5. Aggregate security: a general aggregate signature scheme is aggregate unforgeable, if it holds that

$$\Pr\left[\mathrm{Very}\left((\mathrm{pk}_1, m_1), \ldots, (\mathrm{pk}_n, m_n), \sigma\right) = 1\right] < \mathrm{negl}(\lambda), \quad (3)$$

where the probability is over the experiment $(\mathrm{pk}, \mathrm{sk}) \longleftarrow \mathrm{AS.KeyGen}(1^\lambda)$, $((\mathrm{pk}_1, m_1), \ldots, (\mathrm{pk}_n, m_n), \sigma) \longleftarrow \mathscr{A}^{\mathrm{AS.Sign}(\mathrm{sk}, \cdot)}(\mathrm{pk})$.

Lemma 2 (see [34]). *A general aggregate signature scheme can be constructed based on the difficulty of coCDH problem.*

3. Distributed Functional Signature with Function Privacy

In this section, we give a detailed study of distributed functional signature (DFS), n-out-of-n threshold functional signature. In an (n, n) – DFS scheme, during the key generation algorithm, the secret key corresponding to the function is split into n secret key shares $\left\{(\mathrm{sk}_f^i, f_i)\right\}_{i=1}^n$. Then, we can obtain a pair shared signature $(f_i(m), \sigma_i)$ by running partial signature algorithm on the secret key share sk_f^i corresponding to the shared function f_i and a message m. There is also a reconstruction algorithm that outputs a pair signature $(f(m), \sigma)$ on n shared signatures $\{(f_i(m), \sigma_i)\}_{i=1}^n$.

3.1. Syntax and Security Definition. We present a formal definition of a distributed functional signature, specifying the properties of unforgeability and function privacy. A distributed functional signature scheme for a message space $\mathscr{M}$ and function family $\mathscr{F} = \left\{f: \mathscr{D}_f \longrightarrow \mathscr{M}\right\}$ consists of algorithms (DFS.Setup, DFS.KeyGen, DFS.Sign, DFS.Con, DFS.Very).

(i) $\mathrm{DFS.Setup}(1^\lambda, n) \longrightarrow (\mathrm{msk}, \mathrm{mvk})$: on inputting the secure parameter λ and threshold parameter n, the setup algorithm outputs the master verifiable key mvk and the secret key msk.

(ii) $\mathrm{DFS.KeyGen}(\mathrm{msk}, f) \longrightarrow \left\{(\mathrm{sk}_f^i, f_i)\right\}_{i=1}^n$: on inputting the master secret key msk and a function $f \in \mathscr{F}$, the key generation algorithm outputs n secret key shares $\left\{(\mathrm{sk}_f^i, f_i)\right\}_{i=1}^n$ for the function f.

(iii) $\mathrm{DFS.Sign}(f_i, \mathrm{sk}_f^i, m) \longrightarrow (f_i(m), \sigma_i)$: on inputting the signing key sk_f^i for a function f_i and an input $m \in \mathscr{D}_f$, the signature algorithm outputs a value $f_i(m)$ and a signature σ_i.

(iv) $\mathrm{DFS.Con}(\mathrm{mvk}, \{(f_i(m), \sigma_i)\}_{i=1}^n) \longrightarrow (f(m), \sigma)$: on inputting the master verifiable key mvk and signing pairs $\{(f_i(m), \sigma_i)\}_{i=1}^n$ for the same function f, the reconstruction algorithm outputs a signing pair $(f(m), \sigma)$.

(v) $\mathrm{DFS.Very}(\mathrm{mvk}, m^*, \sigma) \longrightarrow \{0, 1\}$: on inputting the master verifiable key mvk, a message m^*, and a signature σ, the verifiable algorithm outputs "1" or "0," where 1 implies that the signature is valid.

Definition 6. Correctness: an (n, n) – DFS scheme is correct if for any $\mathrm{DFS.Setup}(1^\lambda, n) \longrightarrow (\mathrm{msk}, \mathrm{mvk})$, any $f \in \mathscr{F}$, and any $m \in \mathscr{D}_f$, $\left\{(\mathrm{sk}_f^i, f_i)\right\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f)$, it holds that

$$\begin{aligned} \Pr\left[f(m) = f'(m)\right] &\geq 1 - \mathrm{negl}(\lambda), \\ \Pr\left[\mathrm{DFS.Very}\left(\mathrm{mvk}, f'(m), \sigma\right) = 1\right] &\geq 1 - \mathrm{negl}(\lambda), \end{aligned} \quad (4)$$

where $(f_i(m), \sigma_i) \longleftarrow \mathrm{DFS.Sign}(f_i, \mathrm{sk}_f^i, m)$, $\mathrm{DFS.Con}(\mathrm{mvk}, \{(f_i(m), \sigma_i)\}_{i=1}^n) = (f'(m), \sigma)$, and the probability is taken over the coins in algorithms $\mathrm{DFS.Setup}(1^\lambda, n)$ and $\mathrm{DFS.KeyGen}(\mathrm{msk}, f)$.

Definition 7. Unforgeability: the scheme is unforgeable if the successful probability of any PPT adversary $\mathscr{A}$ in the following game is negligible:

(i) The challenger generates $\mathrm{DFS.Setup}(1^\lambda, n) \longrightarrow (\mathrm{msk}, \mathrm{mvk})$ and sends mvk to $\mathscr{A}$.

(ii) The adversary is allowed to query a key generation oracle and a signing oracle. The challenger initializes a dictionary indexed by tuples $(f, j) \in \mathscr{F} \times \mathscr{N}$, which contains signing keys $\left\{(\mathrm{sk}_f^i, f_i)\right\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f)$.

(iii) Key generation oracle: on inputting (f, j), the challenger runs as follows:

(1) If there exists an entry for the key (f, j) in the dictionary, then output the corresponding value $\left\{(\mathrm{sk}_{fj}^i, f_i^j)\right\}_{i=1}^n$.

(2) Else, run the key generation algorithm $\left\{(\mathrm{sk}_{fj}^i, f_i^j)\right\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f^j)$, add an entry $(f, j) \longrightarrow \left\{(f_i^j, \mathrm{sk}_{fj}^i)\right\}_{i=1}^n$ to the dictionary, and output $\left\{(f_i^j, \mathrm{sk}_{fj}^i)\right\}_{i=1}^n$.

(iv) Signing oracle: on inputting (f, j, m), the challenger runs as follows:

(1) If there exists an entry for the key (f, j) in the dictionary, then run $\left\{\mathrm{DFS.Sign}(f_i^j, \mathrm{sk}_{fj}^i, m) \longrightarrow (f_i^j(m), \sigma_i^j)_{i=1}^n\right\}$ and $\mathrm{DFS.Con}(\mathrm{mvk}, \left\{(f_i^j(m), \sigma_i^j)\right\}_{i=1}^n) \longrightarrow (f^j(m), \sigma^j)$ and output $(f^j(m), \sigma^j)$.

(2) Else, run the key generation algorithm $\left\{(\mathrm{sk}_{fj}^i, f_i^j)\right\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f^j)$, add

an entry $(f, j) \longrightarrow (\{\mathrm{sk}_{f^j}^i\}_{i=1}^n, f_i^j)$ to the dictionary, and generate a signature σ^j on $f^j(m)$ by the following steps: $\{\mathrm{DFS.Sign}(f_i^j, \mathrm{sk}_{f^j}^i, m) \longrightarrow (f_i^j(m), \sigma_i^j)\}_{i=1}^n$, $\mathrm{DFS.Con}(\mathrm{mvk}, \{(f_i^j(m), \sigma_i^j)\}_{i=1}^n) \longrightarrow (f^j(m), \sigma^j)$.

(v) The adversary $\mathscr{A}$ succeeds if it can produce (m^*, σ^*) such that

(1) $\mathrm{DFS.Very}(\mathrm{mvk}, m^*, \sigma^*) = 1$.
(2) There does not exist m such that $f^j(m) = m^*$ and $f_i^j(m) = m^*$ for any f_i^j which was contained in the dictionary.
(3) There does not exist a (f, m) pair such that (f, m) was queried to the signing oracle and $m^* = f(m)$.

Definition 8. Ind-based function privacy: intuitively, we require that the successful advantage of any PPT adversary $\mathscr{A}$ in the following game is negligible:

(i) Setup: the challenger generates a key pair $(\mathrm{msk}, \mathrm{mvk}) \longleftarrow \mathrm{DFS.Setup}(1^\lambda, n)$ and sends the verifiable key mvk to the adversary $\mathscr{A}$.

(ii) Key query: proceeding adaptively, the adversary $\mathscr{A}$ submits $f^j \in \mathscr{F}$ to the challenger. The challenger computes $\{(\mathrm{sk}_{f^j}^i, f_i^j)\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f^j)$ and sends $\{(\mathrm{sk}_{f^j}^i, f_i^j)\}_{i=1}^n$ to the adversary $\mathscr{A}$.

(iii) Challenge phase: $\mathscr{A}$ sends the challenge function pair (f^0, f^1, S) which is not queried in the key query phase. The challenger computes $\{(\mathrm{sk}_{f^b}^i, f_i^b)\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, f^b)$ and sends $\{(\mathrm{sk}_{f^b}^i, f_i^b)\}_{i=1}^n$ to the adversary $\mathscr{A}$, where $b \longleftarrow \{0, 1\}$ and S is chosen randomly from $[n]$ such that $|S| = n - 1$.

(iv) Proceeding adaptively, the adversary can query the key generation oracle and evaluation oracle.

(1) Key query: $\mathscr{A}$ sends a function $f^j \in \mathscr{F}$ with a restriction that $f^j \neq f^0$ and $f^j \neq f^1$. The challenger answers the same as previous key query.
(2) Evaluation query: $\mathscr{A}$ sends an input x_k with a restriction that $f^0(x_k) = f^1(x_k)$. For $i \in ([n]/S)$, challenger returns $(f_i(x_k), \sigma_i) \longleftarrow \mathrm{DFS.Sign}(f_i^b, \mathrm{sk}_{f^b}^i, x_k)$.

(v) Guess: finally, the adversary $\mathscr{A}$ outputs a bit b'. The successful advantage of $\mathscr{A}$ is defined as $\mathrm{Adv} = (\Pr[b' = b] - 1/2)$.

4. The Construction of DFS

We are now ready to describe a construction of distributed functional signature scheme. Informally, our scheme works as follows. In the setup algorithm, the master verifiable key contains a public parameter of function secret sharing scheme and a verifiable key of functional signature. It sets the master signing key of functional signature scheme to the master secret key. The key generation algorithm consists of sharing function algorithm and key generation algorithm of functional signature, which generates n constrained signature keys sk_f^i corresponding to the shadow function f_i. In the signing algorithm, it computes signing pairs $(f_i(m), \sigma_i)$ on inputting a message m. When enough signatures had been collected, anyone can reconstruct the function value and verify its correctness by the verifiable algorithm. We give a detailed description below. Let FSS = (FSS.Setup, FSS.ShareGen, FSS.Recon) be a function secret sharing scheme for function ensemble $\mathscr{F}$ and (FS.Setup, FS.KeyGen, FS.Sign, FS.Very) be a functional signature scheme. The construction of DFS = (DFS.Setup, DFS.KeyGen, DFS.Sign, DFS.Con, DFS.Very) is given as follows.

(i) $\mathrm{DFS.Setup}(1^\lambda, n) \longrightarrow (\mathrm{msk}, \mathrm{mvk})$:

(1) Run the setup algorithm of FS and generate a key pair $(\mathrm{FS.mvk}, \mathrm{FS.msk}) \longleftarrow \mathrm{FS.Setup}(1^\lambda)$.
(2) Run the setup algorithm of FSS and generate a public parameter $\mathrm{FSS.pp} \longleftarrow \mathrm{FS.Setup}(1^\lambda, n, \mathscr{F})$.
(3) Set the master verifiable key to be mvk = (FS.mvk,FSS.pp) and the master secret key to be msk = FS.msk.

(ii) $\mathrm{DFS.KeyGen}(\mathrm{msk}, f) \longrightarrow \{(\mathrm{sk}_f^i, f_i)\}_{i=1}^n$:

(1) Run the function sharing algorithm of FSS and get n sharing functions $\{f_i\}_{i=1}^n \longleftarrow \mathrm{FSS.ShareGen}(\mathrm{FSS.pp}, f)$.
(2) Run the key generation algorithm of FS and generate n constrained signature keys $\mathrm{sk}_f^i \longleftarrow \mathrm{FS.KeyGen}(\mathrm{FS.msk}, f_i)$ for $i = 1, \ldots, n$.
(3) Output the signature pairs $\{(\mathrm{sk}_f^i, f_i)\}_{i=1}^n$.

(iii) $\mathrm{DFS.Sign}(f_i, \mathrm{sk}_f^i, m) \longrightarrow (f_i(m), \sigma_i)$: given the i–th secret key (f_i, sk_f^i), it computes and outputs $(f_i(m), \sigma_i) = \mathrm{FS.Sign}(f_i, \mathrm{sk}_f^i, m)$.

(iv) $\mathrm{DFS.Con}(\mathrm{mvk}, \{(f_i(m), \sigma_i)\}_{i=1}^n) \longrightarrow (f(m), \sigma)$: it computes $f(m) = \mathrm{FSS.Recon}(\mathrm{FSS.pp}, \{f_i(m)\}_{i=1}^n)$ and $\sigma = \{(f_i(m), \sigma_i)\}_{i=1}^n$.

(v) $\mathrm{DFS.Very}(\mathrm{mvk}, m^*, \sigma) \longrightarrow \{0, 1\}$:

(1) If $\sigma = \{(f_i(m), \sigma_i)\}_{i=1}^n$, then it runs $\mathrm{FS.Very}(\mathrm{FS.mvk}, f_i(m), \sigma_i)$ for $i = 1, \ldots, n$. If all of the verifiable algorithms output "1" and $m^* = \mathrm{FSS.Recon}(\mathrm{FSS.pp}\{f_i(m)\}_{i=1}^n)$, it outputs "1"; else, it outputs "0."
(2) If not, it runs $\mathrm{FS.Very}(\mathrm{FS.mvk}, m^*, \sigma)$ and outputs the corresponding result.

Remark 1. In our verifiable algorithm, it can verify two forms of signature. One is generated by the reconstruction algorithm. The another is generated by the signing algorithm.

Correctness. The correctness of our DFS construction follows from the correctness of FS and FSS. Firstly, the outputs $f_i(m)$ of signature algorithm satisfy $f(m) = \mathrm{FSS.Recon}(\mathrm{FSS.pp}, \{f_i(m)\}_{i=1}^n)$ by the reconstruction algorithm

of FSS. Secondly, the verifiable algorithm will output "1" if the signature is generated correctly by the FS scheme.

Theorem 1. *Let FS be an unforgeability functional signature scheme; then, our construction of DFS described above is secure (c.f. Definition 7).*

Proof. We now prove that if there exists a PPT adversary $\mathscr{A}$ that can break the unforgeable security of DFS with non-negligible probability, then another adversary $\mathscr{B}$ can be constructed to break the unforgeable security of FS with the same probability.

Let $\mathscr{C}$ be the challenger for the FS scheme. We construct the adversary $\mathscr{B}$ by $\mathscr{A}$ as follows:

(i) $\mathscr{B}$ first honestly runs the secure experiment of FS and gets FS.mvk of functional signature. Then, it generates FSS.pp$\longleftarrow$FSS.Setup$(1^\lambda, n)$. The adversary $\mathscr{B}$ returns mvk = (FS.mvk,FSS.pp) to $\mathscr{A}$.

(ii) $\mathscr{B}$ initializes a dictionary indexed by tuples $(f, j) \in \mathscr{F} \times \mathscr{N}$, which contains signing keys $\{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$.

(iii) Key generation oracle: on inputting (f, j), $\mathscr{A}$ operates as follows:

(1) If there exists an entry for the key (f, j) in the dictionary, then output the corresponding pair $\{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$.

(2) Else, compute $\{f^j_i\}^n_{i=1} \longleftarrow$FSS.ShareGen (FSS.pp, f^j). For $i = 1, \ldots, n$, $\mathscr{B}$ queries the key generation oracle of FS and gets n secret keys $\{\mathrm{sk}^i_{f^j}\}^n_{i=1}$. Add an entry $(f, j) \longrightarrow \{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$ to the dictionary and output $\{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$.

(iv) Signing oracle: on inputting (f, j, m), $\mathscr{A}$ operates as follows:

(1) If there exists an entry for the key (f, j) in the dictionary, then run $\{\mathrm{DFS.Sign}\ (f^j_i, \mathrm{sk}^i_{f^j}, m) \longrightarrow (f^j_i(m), \sigma^j_i)^n_{i=1}\}$ and DFS.Con(mvk, $\{(f^j_i(m), \sigma^j_i)\}^n_{i=1}) \longrightarrow (f^j(m), \sigma^j)$ and output $(f^j(m), \sigma^j)$.

(2) Else, compute $\{f^j_i\}^n_{i=1} \longleftarrow$FSS.ShareGen (pp, f^j). For $i = 1, \ldots, n$, $\mathscr{B}$ queries the signing oracle $\{(f^j_i, m)\}^n_{i=1}$ of FS and gets n signature pairs $\{(f^j_i(m), \sigma_i)\}^n_{i=1}$. Then, it computes DFS.Con(mvk, $\{(f^j_i(m), \sigma^j_i)\}^n_{i=1}) \longrightarrow (f^j(m), \sigma^j)$ and returns $(f^j(m), \sigma^j)$ to $\mathscr{A}$.

(v) The adversary $\mathscr{A}$ produces a forgeable signature (m^*, σ^*). Then, B computes as follows:

(1) If σ^* is the form of $\{(f^j_i(m), \sigma_i)\}^n_{i=1}$, then $\mathscr{B}$ returns $(f^j_1(m), \sigma^j_1)$ to the challenger.

(2) Else, $\mathscr{B}$ returns (m^*, σ^*) to the challenger.

We observe that $\mathscr{B}$ perfectly simulates the unforgeable experiment of DFS. If $\mathscr{A}$ successfully outputs a forgeable signature (m^*, σ^*), it must satisfy the following:

(1) DFS.Very$(\mathrm{mvk}, m^*, \sigma^*) = 1$.

(2) There does not exist m such that $f^j(m) = m^*$ and $f^j_i(m) = m^*$ for any f^j_i which was contained in the dictionary.

(3) There does not exist a (f, m) such that (f, m) was a query to the signing oracle and $m^* = f(m)$.

If the forgeable signature (m^*, σ^*) is the first form, then FS.Very(FS.mvk, $f_i(m), \sigma_i) = 1$ for any $i = 1, \ldots, n$. Because f_i is not contained in the dictionary and (m^*, σ^*) is not queried to the signature oracle, $(f^j_1(m), \sigma^j_1)$ is a legally forgeable signature for the functional signature scheme. If the forgeable signature (m^*, σ^*) is the second form, then (m^*, σ^*) is a distributive signature and FS.Very(FS.mvk, $m^*, \sigma^*) = 1$. There does not exist m such that $f_i(m) = m^*$ which is contained in the dictionary; it is also a legally forgeable signature. Therefore, if $\mathscr{A}$ can break the unforgeability security of DFS with non-negligible probability, then we can construct another adversary $\mathscr{B}$ which breaks the unforgeability security of FS with the same probability. □

Theorem 2. *Assume FSS.Con is an invertible algorithm. FSS.Con is an invertible algorithm if for any $n-1$ inputs $\{f_i(x)\}^{n-1}_{i=1}$ and an output $f(x)$, it can compute the n–th input $f_n(x)$, where the functions $\{f_i(x)\}^{n-1}_{i=1}$ share function of $f(x)$. For example, Fan and Tang [12] gave a function secret sharing in which the reconstruction algorithm is $f(x) = \sum^n_{i=1} f_i(x)$, and $f_n(x) = f(x) - \sum^{n-1}_{i=1} f_i(x)$. Let FSS be a secure function secret sharing scheme (c.f. Definition 2); then, our construction of DFS described above satisfies function privacy (c.f. Definition 8).*

Proof. We now prove that if there exists a PPT adversary $\mathscr{A}$ that can break the function privacy of DFS with non-negligible probability, then another adversary $\mathscr{B}$ can be constructed to break the security of FSS with the same probability.

Let $\mathscr{C}$ be the challenger for the FSS scheme. We construct the adversary $\mathscr{B}$ by $\mathscr{A}$ as follows:

(i) $\mathscr{B}$ first honestly runs the secure experiment of FSS and gets FSS.pp of function secret sharing scheme. Then, it generates (FS.mvk, FS.msk)$\longleftarrow$FS.Setup(1^λ). The adversary $\mathscr{B}$ returns mvk = (FSS.pp, FS.mvk) to $\mathscr{A}$.

(ii) $\mathscr{B}$ initializes a dictionary indexed by tuples $(f, j) \in \mathscr{F} \times \mathscr{N}$, which contains signing keys $\{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$.

(iii) Key query: on inputting (f, j), $\mathscr{B}$ operates as follows:

(1) If there exists an entry for the key (f, j) in the dictionary, then output $\{(\mathrm{sk}^i_{f^j}, f^j_i)\}^n_{i=1}$.

(2) Else, compute $\{f_i^j\}_{i=1}^n \longleftarrow$FSS.ShareGen (FSS.pp, f^j). For $i = 1, \ldots, n$, $\mathscr{B}$ computes $\{(\mathrm{sk}_{f^j}^i, f_i^j)\} \longleftarrow$FS.KeyGen (FS.msk, f_i^j). Add an entry $(f, j) \longrightarrow \{(\mathrm{sk}_{f^j}^i, f_i^j)\}_{i=1}^n$ to the dictionary and output $\{(\mathrm{sk}_{f^j}^i, f_i^j)\}_{i=1}^n$.

(iv) $\mathscr{A}$ sends the challenge function pair (f^0, f^1, S) to the adversary $\mathscr{B}$, where both f^0 and f^1 were not queried in the previous phase, and $D_{f^0} = D_{f^1}, |S| = n - 1$. Then, $\mathscr{B}$ queries the challenger of FS on (f^0, f^1, S) and gets $\{f_i^b\}_{i \in S}$. For $i \in S$, $\mathscr{B}$ computes $\{(\mathrm{sk}_{f^b}^i, f_i^b)\} \longleftarrow$FS.KeyGen (FS.msk, f_i^b) and returns $\{(\mathrm{sk}_{f^b}^i, f_i^b)\}_{i \in S}$ to the adversary $\mathscr{A}$.

(v) Proceeding adaptively, $\mathscr{A}$ can query the key generation oracle and evaluation oracle.

(1) Key query: $\mathscr{A}$ sends a function $f^j \in \mathscr{F}$ with a restriction that $f^j \neq f^0$ and $f^j \neq f^1$. $\mathscr{B}$ answers the same as previous key query.

(2) Evaluation query: $\mathscr{A}$ sends an input x_k with a restriction that $f^0(x_k) = f^1(x_k)$. For $i \in ([n]/S)$, $\mathscr{B}$ first computes $\{(f_i^b(x_k), \sigma_i)\}_{i \in S}$ and $f^0(x_k)$. Then, $\mathscr{B}$ computes $f_{i \in ([n]/S)}(x_k)$ by the invertible property of FSS.Con and a corresponding signature $\sigma_{i \in ([n]/S)}$ by the master secret key FS.msk of functional signature. For a functional signature scheme, it has a master signing key FS.msk, which can be used to sign any message. At the same time, it also has a constrained key sk_f for the function f, which only signs the message in the range of f. $\mathscr{B}$ returns $(f_{i \in ([n]/S)}(x_k), \sigma_{i \in ([n]/S)})$ to $\mathscr{A}$.

(vi) At last, the adversary $\mathscr{B}$ outputs the guess b' of the adversary $\mathscr{A}$.

Although the generation way of the signature $\sigma_{i \in ([n]/S)}$ is changed, it does not influence the view of $\mathscr{A}$ because this signature can be verified by the verifiable algorithm of FS. Therefore, $\mathscr{B}$ perfectly simulates function private experiment. If $\mathscr{A}$ can distinguish the function f^0 and f^1 with nonnegligible advantage, then $\mathscr{B}$ can break the security of FSS with the same advantage.

In the above construction, the size of combined signature depends on the threshold, which will reduce the verification efficiency. In order to improve the efficiency, it can use aggregate signature to compress the final signature size based on the construction of functional signature proposed by Boyle et al. [9]. In their key generation algorithm, it generates the secret key sk_f for the function f by a standard signature scheme. Therefore, we can change the standard signature into an aggregate signature. The generated signatures $\{\sigma_1, \ldots, \sigma_n\}$ of functional signature scheme can be aggregated into a single signature σ. This change does not affect the security of the construction because the security of Boyle's construction is reduced to the unforgeability of a standard signature. On the one hand, it does not use the aggregate property. On inputting an aggregate signature, the verifiable algorithm of functional signature will output one bit "0." On the other hand, if the adversary can produce a forgeable signature for functional signature scheme by the aggregate signature, then it can be used to break the aggregate security. Based on the first construction, we modify the reconstruction algorithm and verifiable algorithm. The specific description is as follows.

(i) DFS.Setup $(1^\lambda, n) \longrightarrow$ (msk, mvk): it runs the same as before.

(ii) DFS.KeyGen (msk, f) $\longrightarrow \{(\mathrm{sk}_f^i, f_i)\}_{i=1}^n$: it runs the same as before.

(iii) DFS.Sign $(f_i, \mathrm{sk}_f^i, m) \longrightarrow (f_i(m), \sigma_i)$: it runs the same as before.

(iv) DFS.Con (mvk, $\{(f_i(m), \sigma_i)\}_{i=1}^n) \longrightarrow (f(m), \sigma)$: it computes $f(m) =$ FSS.Recon (FSS.pp$\{f_i(m)\}_{i=1}^n$) and $\sigma' =$ AS.Agg $(\{(f_i(m), \sigma_i)\}_{i=1}^n)$ and returns $(f(m), \sigma = (\sigma', \{f_i(m)\})_{i=1}^n)$.

(v) DFS.Very (mvk, $m^*, \sigma) \longrightarrow \{0, 1\}$:

(1) If $\sigma = (\sigma', \{f_i(m)\}_{i=1}^n)$, then it runs AS.Very $(\sigma', \{f_i(m)\}_{i=1}^n)$. If the verifiable algorithms output '1' and $m^* =$ FSS.Recon (FSS.pp, $\{f_i(m)\}_{i=1}^n$), it outputs "1"; otherwise, it outputs "0."

(2) If not, it runs FS.Very (FS.mvk, m^*, σ) and outputs the corresponding result.

The secure proofs of the modified construction are similar to the Theorems 1 and 2. In the proof of unforgeability, there are two forms of signature. The first one is $(m^*, \sigma = (\sigma', \{f_i(m^*)\}_{i=1}^n))$ which satisfies AS.Very $(\sigma', \{f_i(m^*)\}_{i=1}^n) = 1$. Because the reconstruction algorithm is run honestly, there exists an index $i \in \{1, \ldots, n\}$ such that FS.Very (FS.mvk, $f_i(m^*), \sigma_i^*) = 1$. Meanwhile, it satisfies that f_i is not contained in the dictionary, and (m^*, σ^*) is not queried to the signature oracle. So, $(f_i(m^*), \sigma_i^*)$ is a legally forgeable signature for the functional signature scheme. The second form of signature is the same as Theorem 1, which can also be reduced to the security of functional signature. The proof of function privacy is exactly the same as Theorem 2. □

5. Hosting Services Securely in Multiple Untrusted Clouds

We consider the setting of hosting service in untrusted clouds which was first presented by Boneh [11]. In this model, it has three parties: service provider, who owns a program P, cloud server, who provides the computation capability, and arbitrary many clients. The service provider wants to host the program P on a cloud server and authenticate the clients who pay for the service provided by program P. Only the authenticated clients can access the program hosted on the cloud server and compute output on inputs of their choice. Meanwhile, the program P may contain proprietary information, and it needs to protect its

privacy. Moreover, the scheme should satisfy some properties [11]:

(i) Weak client: the total cost of the client should only depend on the size of input and security parameter and should be independent of the running time of program P.

(ii) Delegation: the service provider only needs to run one-time setup of the whole system and authentication clients. The total cost should be bounded by a fixed polynomial in the program size in the setup phase and should only depend on the security parameter in the authentication phase.

(iii) Polynomial slowdown: the running time of the cloud is bounded by a fixed polynomial in the running time of the program P.

In our verifiable distributed secure cloud service scheme, the service provider generates a set of encoded program shares for program P and then hosts each encoded program share on one cloud server. Any authenticated client can access the encoded program shares hosted in multiple cloud servers and compute output on inputs of his choice. Meanwhile, we require the client to verify the correctness of returned result from the cloud servers.

5.1. Syntax and Security Definitions. We recall the notion of secure cloud service scheme proposed by Boneh et al. [11] and make some changes based on [12]. The verifiable distributed secure cloud service scheme consists of five algorithms VDS = (VDS.Prog, VDS.Auth, VDS.Inp, VDS.Eval, VDS.Very) which is described as follows:

(i) $\mathrm{VDS.Prog}(1^\lambda, n, P) \longrightarrow \{(\{P_i\}_{i=1}^n, \mathrm{sk})\}$: on inputting the security parameter λ, the threshold parameter n, and a program P, the program generation algorithm outputs encoded programs $\{P_i\}_{i=1}^n$ and a secret key sk which is used in the authentication algorithm.

(ii) $\mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id}) \longrightarrow \mathrm{token}_{\mathrm{id}}$: on inputting an identity id of a client and the secret key sk, the authentication algorithm outputs a token $\mathrm{token}_{\mathrm{id}}$ for the client.

(iii) $\mathrm{VDS.Inp}(\mathrm{token}_{\mathrm{id}}, x) \longrightarrow (\widehat{x}, \alpha)$: on inputting the token $\mathrm{token}_{\mathrm{id}}$ and an input x, this algorithm outputs an encoded input $\widehat{x}$ and α which is used by client to verify the evaluated value.

(iv) $\mathrm{VDS.Eval}(P_i, \widehat{x}) \longrightarrow \widehat{y}_i$: on inputting the encoded program P_i and input $\widehat{x}$, the evaluation algorithm outputs a result $\widehat{y}_i = P_i(\widehat{x})$.

(v) $\mathrm{VDS.Very}(\alpha, \{\widehat{y}\}_{i=1}^n) \longrightarrow P(x)$ or $\perp$: on inputting the verifiable information α and the evaluated value $\widehat{y}_i$, the verifiable algorithm outputs the value $P(m)$ or $\perp$, which implies that the client rejects the evaluated result.

The procedure runs as follows: the server provider first runs the algorithm $\mathrm{VDS.Prog}(1^\lambda, n, P)$ and obtains the distributed encoded program $\{P_i\}_{i=1}^n$ and a secret key sk. Then, it sends P_i to the ith cloud server. If a client wants to access the program hosted on the cloud server, the service provider would authenticate the client by the secret key sk. An authenticated client with identity id can encode its inputs by the algorithm $\mathrm{VDS.Inp}(\mathrm{token}_{\mathrm{id}}, x)$ and send $\widehat{x}$ to each cloud servers, respectively. Every cloud server will evaluate the sharing program P_i on encoded input $\widehat{x}$ and return the evaluation $P_i(\widehat{x})$. Finally, the client can reconstruct the value $P(x)$ by the algorithm $\mathrm{VDS.Very}(\alpha, \{\widehat{y}\}_{i=1}^n)$ and verify its correctness. In Figure 1, we give the algorithm flowchart for the verifiable distributed secure cloud service scheme.

Two cases for security definition are considered in [11], untrusted cloud security and untrusted client security. We consider the case of untrusted cloud security into two subcases, program privacy and verifiability.

Definition 9 (untrusted cloud security: program privacy) (see [12], Definition 4.1). A VDS scheme is program privacy in untrusted cloud security if the successful advantage of any PPT adversary $\mathscr{A}$ in the following experiment is negligible:

(i) The adversary $\mathscr{A}$ sends challenge program (P_0, P_1, S) to challenger such that $S \subset [n]$, and $|S| = n - 1$. The challenger samples a random bit $b \longleftarrow \{0, 1\}$, computes the encoded program $(\{P_i^b\}_{i=1}^n, \mathrm{sk}) \longleftarrow \mathrm{VDS.Prog}(1^\lambda, n, P_b)$, and sends $\{P_i^b\}_{i \in S}$ to adversary $\mathscr{A}$.

(ii) Proceeding adaptively, the adversary $\mathscr{A}$ can make the authentication query and input query:

(1) Authentication query: the challenger initializes a dictionary indexed by $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$. When $\mathscr{A}$ queries an identity id_j, the challenger returns $\mathrm{token}_{\mathrm{id}_j}$ to adversary $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, the challenger returns $\mathrm{token}_{\mathrm{id}_j} \longleftarrow \mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id})$ and updates the dictionary $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$.

(2) Input query: A sends (id, j, x) to the challenger. If there exists $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$ in the dictionary, challenger returns $(\widehat{x}, \alpha) \longleftarrow \mathrm{VDS.Inp}(\mathrm{token}_{\mathrm{id}_j}, x)$. If not, challenger computes $\mathrm{token}_{\mathrm{id}_j} \longleftarrow \mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id}_j)$, returns $(\widehat{x}, \alpha) \longleftarrow \mathrm{VDS.Inp}(\mathrm{token}_{\mathrm{id}_j}, x)$, and updates the dictionary $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$.

(iii) Finally, the adversary $\mathscr{A}$ outputs his guess b' for the bit b. The adversary $\mathscr{A}$ succeeds if $b' = b$. The advantage of $\mathscr{A}$'s success is defined as $(\Pr[b' = b] - 1/2)$.

Definition 10 (untrusted cloud security: verifiability). A VDS scheme is verifiable in untrusted cloud security if the successful probability of any PPT adversary $\mathscr{A}$ in the following experiment is negligible:

(i) The adversary $\mathscr{A}$ sends challenge program P to challenger. The challenger samples the encoded

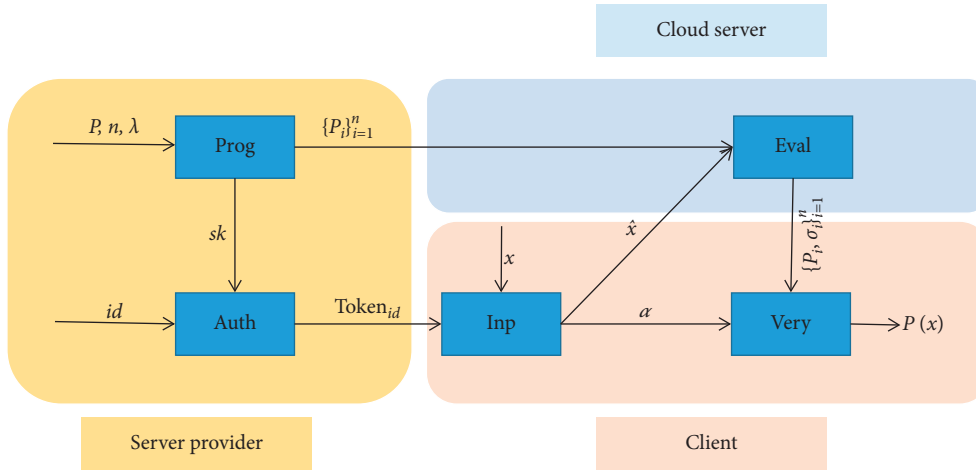


Figure 1: Algorithm flowchart for VDS.

program $(\{P_i\}_{i=1}^n, \mathrm{sk}) \longleftarrow \mathrm{VDS.Prog}(1^\lambda, n, P)$ and sends $\{P_i\}_{i=1}^n$ to adversary $\mathscr{A}$.

(ii) Proceeding adaptively, the adversary $\mathscr{A}$ can make the authentication query. The challenger initializes a dictionary indexed by $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$. When $\mathscr{A}$ queries an identity id_j, the challenger returns $\mathrm{token}_{\mathrm{id}_j}$ to adversary $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, the challenger returns $\mathrm{token}_{\mathrm{id}_j} \longleftarrow \mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id})$ and updates the dictionary $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$.

(iii) Finally, $\mathscr{A}$ outputs $(x, \alpha, \{\widehat{y}\}_{i=1}^n)$. $\mathscr{A}$ succeeds if $P(x) \neq \mathrm{VDS.Very}(\alpha, \{\widehat{y}\}_{i=1}^n)$.

For untrusted client security, we require that a collection of corrupt clients with the help of a subset of corrupt servers does not distinguish which program is encoded.

Definition 11 (untrusted client security) (see [12], Definition 4.4). The VDS scheme is untrusted client security if the successfully advantage of any PPT adversary $\mathscr{A}$ in the following experiment is negligible:

(i) The adversary $\mathscr{A}$ sends challenge program (P_0, P_1, S) to challenger such that $S \subset [n]$, $|S| = n - 1$. The challenger samples a random bit $b \longleftarrow \{0, 1\}$, computes the encoded program $(\{P_i^b\}_{i=1}^n, \mathrm{sk}) \longleftarrow \mathrm{VDS.Prog}(1^\lambda, n, P_b)$, and sends $\{P_i^b\}_{i \in S}$ to adversary $\mathscr{A}$.

(ii) Proceeding adaptively, the adversary $\mathscr{A}$ can make the authentication query and evaluation query:

(1) Authentication query: the challenger initializes a dictionary indexed by $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$. When $\mathscr{A}$ queries an identity id_j, the challenger returns $\mathrm{token}_{\mathrm{id}_j}$ to adversary $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, the challenger returns $\mathrm{token}_{\mathrm{id}_j} \longleftarrow \mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id})$ and updates the dictionary $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$.

(2) Evaluation query: A sends an encoding $\widehat{x}_k$ for the input x_k to the challenger such that $P_0(x_k) = P_1(x_k)$. For $i \in ([n]/S)$, challenger returns $\widehat{y}_{i,k} = P_i^b(\widehat{x}_k)$.

(iii) Finally, the adversary $\mathscr{A}$ outputs his guess b' for the bit b. The adversary $\mathscr{A}$ succeeds if $b' = b$. The successful advantage of $\mathscr{A}$ is defined as $(\Pr[b' = b] - 1/2)$.

5.2. Our VDS Construction. Let VDS = (VDS.Prog, VDS.Auth, VDS.Inp, VDS.Eval, VDS.Very) be a distributed functional signature scheme and Sig = (Sig.Setup, Sig.Sign, Sig.Very) be an existential unforgeable signature scheme. Now, we describe our construction as follows:

(i) $\mathrm{VDS.Prog}(1^\lambda, n, P) \longrightarrow (\{P_i\}_{i=1}^n, \mathrm{sk})$:

(1) Run $(\mathrm{msk}, \mathrm{mvk}) \longleftarrow \mathrm{DFS.Setup}(1^\lambda, n)$ and $(\mathrm{Sig.sk}, \mathrm{Sig.vk}) \longleftarrow \mathrm{Sig.Setup}(1^\lambda)$.

(2) Construct a new program $\widehat{P}$, which is given in Figure 2.

(3) Compute $\left\{(\mathrm{sk}_{\widehat{P}}^i, \widehat{P}_i)\right\}_{i=1}^n \longleftarrow \mathrm{DFS.KeyGen}(\mathrm{msk}, \widehat{P})$.

(4) Set $P_i = (\mathrm{sk}_{\widehat{P}}^i, \widehat{P}_i)$ and $\mathrm{sk} = (\mathrm{Sig.sk}, \mathrm{mvk})$.

(ii) $\mathrm{VDS.Auth}(\mathrm{sk}, \mathrm{id}) \longrightarrow \mathrm{token}_{\mathrm{id}}$: parse $\mathrm{sk} = (\mathrm{Sig.sk}, \mathrm{mvk})$, compute the signature $\sigma_{\mathrm{id}} = \mathrm{Sig.Sign}(\mathrm{Sig.sk}, \mathrm{id})$, and output $\mathrm{token}_{\mathrm{id}} = (\mathrm{mvk}, \sigma_{\mathrm{id}}, \mathrm{id})$.

(iii) $\mathrm{VDS.Inp}(\mathrm{token}_{\mathrm{id}}, x) \longrightarrow (\widehat{x}, \alpha)$: parse $\mathrm{token}_{\mathrm{id}} = (\mathrm{mvk}, \sigma_{\mathrm{id}}, \mathrm{id})$ and set $\widehat{x} = (x, \sigma_{\mathrm{id}}, \mathrm{id})$ and $\alpha = (x, \mathrm{mvk})$.

(iv) $\mathrm{VDS.Eval}(P_i, \widehat{x}) \longrightarrow \widehat{y}_i$: parse $P_i = (\mathrm{sk}_{\widehat{P}}^i, \widehat{P}_i)$ and compute $\widehat{y}_i = (\widehat{P}_i(x), \sigma_i) \longleftarrow \mathrm{DFS.Sign}(\widehat{P}_i^i, \mathrm{sk}_{\widehat{P}}^i, \widehat{x})$.

(v) $\mathrm{VDS.Very}(\alpha, \{\widehat{y}_i\}_{i=1}^n) \longrightarrow P(x)$ or $\perp$: parse $\alpha = (x, \mathrm{mvk})$. Compute $\mathrm{DFS.Con}(\mathrm{mvk}, \{(\widehat{P}_i(x), \sigma_i)\}_{i=1}^n) = (\widehat{P}(x), \sigma)$. Parse $\widehat{P}(x) = x' | P'(x)$. If

Input: signature σ, input x and identity id.
Constants: sig.vk and program P.

(a) If sig.very (sig.vk, σ, id) = 0, output $\perp$.
(b) Else, compute and output $x|P(x)$.

Figure 2: Program $\widehat{P}$.

DFS.Very$(\text{mvk}, \widehat{P}(x), \sigma) = 1$ and $x' = x$, output $P'(x)$. Else, it rejects the returned results.

Correctness. The correctness of our VDS construction follows from the correctness of underlying distributed functional signature scheme DFS and the signature scheme Sig. We can observe that it outputs $\perp$ for an invalid signature in the encoded program $\widehat{P}_i$. Otherwise, it outputs $\widehat{y}_i = (\widehat{P}_i(x), \sigma_i)$. By the correctness of DFS, the output of VDS.Very$(\alpha, \{\widehat{y}_{i=1}^n\})$ is $P(x)$.

Efficiency. We do not consider the time cost of algorithm VDS.Prog because the cost of encoding the program can be amortized over many input computations. For each invocation of VDS.Auth, the service provider generates a signature for the client id. Therefore, it is efficient for the service provider. The work of the client in handling his inputs only depends polynomially on the size of inputs and a security parameter. For the verifiable algorithm, the client needs to reconstruct the function evaluation and verify a signature, which is independent of the complexity of the program. In the phase of reconstructing the function evaluation, the client just needs to compute a summation operation by using the function secret sharing proposed by Fan and Tang [12].

Security. We show that our DFS construction satisfies untrusted cloud security (program privacy and verifiability) and untrusted client security defined as above.

Theorem 3. *Let DFS be a distributed functional signature scheme against the security of function privacy (c.f. Definition 8); then, our construction of VDS described above has program privacy in untrusted cloud security (c.f. Definition 9).*

Proof. We prove that if there exists a PPT adversary $\mathscr{A}$ that can break the program privacy of VDS with non-negligible probability, then another adversary $\mathscr{B}$ can be constructed to break the function privacy of DFS with the same probability.

Let $\mathscr{C}$ be the challenger for the DFS scheme. We construct the adversary $\mathscr{B}$ by $\mathscr{A}$ as follows:

(i) The adversary $\mathscr{A}$ sends challenge program (P_0, P_1, S) to $\mathscr{B}$ such that $S \subset [n], |S| = n - 1$. $\mathscr{A}$ samples a signing key $(\text{Sig.vk}, \text{Sig.sk}) \longleftarrow \text{Sig.Setup}(1^\lambda)$, constructs two new programs $(\widehat{P}_0, \widehat{P}_1)$ as defined in Figure 2, and sends $(\widehat{P}_0, \widehat{P}_1, S)$ to the challenger of DFS. The challenger returns $(\left\{(\text{sk}_{\widehat{P}}^i, \widehat{P}_i^b)\right\}_{i \in S}, \text{mvk})$. $\mathscr{B}$ returns $\left\{(\text{sk}_{\widehat{P}}^i, \widehat{P}_i^b)\right\}_{i \in S}$ to $\mathscr{A}$ and sets $\text{sk} = (\text{Sig.sk}, \text{mvk})$.

(ii) Proceeding adaptively, the adversary $\mathscr{A}$ can make the authentication query and input query:

(1) Authentication query: $\mathscr{B}$ initializes a dictionary indexed by $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$. When $\mathscr{A}$ queries an identity id_j, $\mathscr{B}$ returns $\text{token}_{\text{id}_j}$ to $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, $\mathscr{B}$ computes $\sigma_{\text{id}_j} \longleftarrow \text{Sig.Sign}(\text{Sig.sk}, \text{id}_j)$, then returns $\text{token}_{\text{id}_j} = (\text{mvk}, \sigma_{\text{id}_j}, \text{id})$, and updates the dictionary $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$.

(2) Input query: $\mathscr{B}$ sends (id, j, x) to the challenger. If there exists $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$ in the dictionary, $\mathscr{A}$ returns $\widehat{x} = (x, \sigma_{\text{id}}, \text{id}), \alpha = (x, \text{mvk})$ to adversary. Else, $\mathscr{A}$ computes $\text{token}_{\text{id}_j} \longleftarrow \text{VDS.Auth}(\text{sk}, \text{id})$, returns $(\widehat{x}, \alpha) \longleftarrow \text{VDS.Inp}(\text{token}_{\text{id}_j}, x)$, and updates the dictionary $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$.

(iii) Finally, the adversary $\mathscr{A}$ outputs the guess b' of $\mathscr{B}$.

We observe that $\mathscr{B}$ perfectly simulates program private experiment of VDS. Therefore, if $\mathscr{B}$ could distinguish the program P_0 and P_1 with non-negligible advantage in untrusted cloud security, then $\mathscr{A}$ can break the security of DFS with the same advantage. □

Theorem 4. *Let DFS be a distributed functional signature scheme against the security of unforgeability (c.f. Definition 7); then, our construction of VDS described above is verifiable in untrusted cloud security (c.f. Definition 10).*

Proof. We prove that if there exists a PPT adversary $\mathscr{A}$ that can break verifiability of VDS with non-negligible probability, then another adversary $\mathscr{B}$ can be constructed to break unforgeability of DFS with the same probability.

Let $\mathscr{C}$ be the challenger for the DFS scheme. We construct the adversary $\mathscr{B}$ by $\mathscr{A}$ as follows:

(i) $\mathscr{B}$ honestly runs the unforgeable experiment of DFS and gets mvk of distributed functional signature. Then, $\mathscr{B}$ samples a signing pair $(\text{Sig.sk}, \text{Sig.vk}) \longleftarrow \text{Sig.Sign}(1^\lambda)$.

(ii) $\mathscr{A}$ sends challenge program P to $\mathscr{B}$. $\mathscr{B}$ defines a new function $\widehat{P}$ which is described in Figure 2 and sends $\widehat{P}$ to the challenger $\mathscr{C}$ of DFS. The challenger returns $\left\{(\widehat{P}_i, \text{sk}_{\widehat{P}}^i)\right\}_{i=1}^n$, sets $P_i = (\widehat{P}_i, \text{sk}_{\widehat{P}}^i)$, and sends $\{P_i\}_{i=1}^n$ to adversary $\mathscr{A}$.

(iii) Proceeding adaptively, $\mathscr{A}$ can make the authentication query. $\mathscr{B}$ initializes a dictionary indexed by $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$. When $\mathscr{A}$ queries an identity id_j, the challenger returns $\text{token}_{\text{id}_j}$ to adversary $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, the challenger returns $\text{token}_{\text{id}_j} = (\sigma_{\text{id}_j}, \text{id}_j, \text{mvk})$ and updates the dictionary $(\text{id}, j) \longrightarrow \text{token}_{\text{id}_j}$, where $\sigma_{\text{id}} = \text{Sig.Sign}(\text{Sig.sk}, \text{id}_j)$.

(iv) Finally, $\mathscr{A}$ outputs $(x, \alpha, \{\widehat{y}_i = (\widehat{P}_i(x), \sigma_i)\}_{i=1}^n)$. Then, $\mathscr{B}$ computes $(\widehat{P}(x), \sigma) \longleftarrow \text{VDS.Very}$ $(\alpha, \{\widehat{y}\}_{i=1}^n)$ and outputs the forgeable signature $(x'|P'(x), \sigma)$, where $\widehat{P}(x) = x'|P'(x)$.

Assume that $\mathscr{A}$ succeeded in the experiment of verifiability; then, it must satisfy that $\mathrm{DFS.Very}(\mathrm{mvk}, \widehat{P}(x), \sigma) = 1$, $x' = x$, and $P'(x) \neq P(x)$. For $i = 1, \ldots, n$, $\mathrm{DFS.Very}(\mathrm{mvk}, \widehat{P}_i(x), \sigma_i) = 1$; then, $x|P_i(x)$ must be contained in the range of program $\widehat{P}_i$. Because the adversary $\mathscr{A}$ can just sign the message belonging to the range of encoded programs $\widehat{P}_i$, the evaluation $x|P_i(x)$ must be computed correctly. In addition, $x|P'(x)$ is not contained in the range of $\widehat{P}$ because of $P'(x) \neq P(x)$. $(x|P'(x), \sigma)$ is validly forgeable signature for the DFS scheme. So, if there exists an adversary that breaks the verifiability of VDS with non-negligible probability, then there exists another adversary that breaks the unforgeability of DFS with the same probability. □

Theorem 5. *Let DFS be a distributed functional signature scheme against the security of function privacy (c.f. Definition 8) and Sig be an unforgeable signature scheme; then, our construction of VDS described above is secure in untrusted client security (c.f. Definition 11).*

Proof. We prove that if there exists a PPT adversary $\mathscr{A}$ that can break security in untrusted client security with non-negligible probability, then another adversary $\mathscr{B}$ can be constructed to break function privacy of DFS with the same probability.

Let $\mathscr{C}$ be the challenger for the DFS scheme. We construct the adversary $\mathscr{B}$ by $\mathscr{A}$ as follows:

(i) $\mathscr{B}$ runs the function private experiment of DFS and gets a master verifiable key mvk of DFS. Then, $\mathscr{B}$ samples a signing pair $(\mathrm{Sig.sk}, \mathrm{Sig.vk}) \longleftarrow \mathrm{Sig.Sign}(1^\lambda)$.

(ii) The adversary $\mathscr{A}$ sends challenge program (P_0, P_1, S) to $\mathscr{B}$ such that $S \subset [n]$, $|S| = n - 1$. $\mathscr{B}$ defines two new functions $\widehat{P}_0$ and $\widehat{P}_1$ which are described in Figure 2 and sends $(\widehat{P}_0, \widehat{P}_1, S)$ to the challenger $\mathscr{C}$ of DFS. The challenger returns $\left\{(\widehat{P}_i^b, \widehat{\mathrm{sk}}_{P_b}^i)\right\}_{i \in S}$, sets $P_i = \left\{(\widehat{P}_i^b, \widehat{\mathrm{sk}}_{P_b}^i)\right\}$, and sends $\{P_i\}_{i \in S}$ to adversary $\mathscr{A}$.

(iii) Proceeding adaptively, the adversary $\mathscr{A}$ can make the authentication query and evaluation query:

  (1) Authentication query: $\mathscr{B}$ initializes a dictionary indexed by $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$. When $\mathscr{A}$ queries an identity id_j, the challenger returns $\mathrm{token}_{\mathrm{id}_j}$ to adversary $\mathscr{A}$ if there exists a (id, j) in the dictionary. If not, the challenger returns $\mathrm{token}_{\mathrm{id}_j} = (\sigma_{\mathrm{id}_j}, \mathrm{id}_j, \mathrm{mvk})$ and updates the dictionary $(\mathrm{id}, j) \longrightarrow \mathrm{token}_{\mathrm{id}_j}$, where $\sigma_{\mathrm{id}} = \mathrm{Sig.Sign}(\mathrm{Sig.sk}, \mathrm{id}_j)$.

  (2) Evaluation query: $\mathscr{A}$ sends an encoding $\widehat{x}_k$ for the input x_k to $\mathscr{B}$. $\mathscr{B}$ first checks whether $P_0(x_k)$ is equal to $P_1(x_k)$. If not, $\mathscr{B}$ stops the experiment. Otherwise, $\mathscr{B}$ queries the evaluation oracle of DFS and returns the output to $\mathscr{A}$.

(iv) Finally, the adversary $\mathscr{B}$ outputs the result b' of the adversary $\mathscr{A}$.

We observe that $\mathscr{B}$ perfectly simulates the secure experiment in untrusted client model. Therefore, if $\mathscr{A}$ can distinguish the program P^0 and P^1 with non-negligible advantage, then $\mathscr{B}$ can break the function private security of DFS with the same advantage. □

6. Conclusion

In this work, we introduce the notion of distributed functional signature. In such model, it provides a possibility of function privacy in the setting of functional signature. We give a construction from function secret sharing and functional signature. The function secret sharing can be constructed based on learning with error assumption, and the functional signature can be constructed based on the existence of one-way function. Therefore, our construction can be obtained from one-way function and learning with error assumption. In addition, we apply it to the host service in multiple untrusted clouds in which the clouds will be dishonest. We require that the returned result from the clouds can be verified.

Data Availability

No underlying data were used in the study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (nos. 61871430 and 61971458) and the Key Technologies R&D Program of Henan Province (nos. 212102210088 and 202102210169).

References

[1] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[2] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM*, vol. 17, no. 2, pp. 281–308, 1988.

[3] D. Chaum, "Blind signature system," in *Advances in Cryptology, Proc. CRYPTO'83*, D. Chaum, Ed., p. 153, Plenum Press, New York, NY, USA, 1984.

[4] D. Chaum and E. van Heyst, "Group signatures," in *Proceedings of the Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques*, pp. 257–265, Brighton, UK, April 1991.

[5] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Proceedings of the Advances in Cryptology-ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 552–565, Gold Coast, Australia, December 2001.

[6] J. Chang, H. Wang, F. Wang, A. Zhang, and Y. Ji, "RKA security for identity-based signature scheme," *IEEE Access*, vol. 8, pp. 17833–17841, 2020.

[7] J. Chang, Y. Ji, B. Shao, M. Xu, and R. Xue, "Certificateless homomorphic signature scheme for network coding," *IEEE/*

ACM Transactions on Networking, vol. 28, no. 6, pp. 2615–2628, 2020.

[8] J. Chang, B. Shao, Y. Ji, M. Xu, and R. Xue, "Secure network coding from secure proof of retrievability," *Science China Information Sciences [Ol], Early Access*, vol. 24, 2021.

[9] E. Boyle, S. Goldwasser, and I. Ivan, "Functional signatures and pseudorandom functions," in *Proceedings of the Public-Key Cryptography-PKC 2014-17th International Conference on Practice and Theory in Public-Key Cryptography*, pp. 501–519, Buenos Aires, Argentina, March 2014.

[10] E. Boyle, N. Gilboa, and Y. Ishai, "Function secret sharing: improvements and extensions," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1292–1303, Vienna, Austria, October 2016.

[11] D. Boneh, D. Gupta, I. Mironov, and A. Sahai, "Hosting services on an untrusted cloud," in *Proceedings of the Advances in Cryptology-EUROCRYPT 2015-34th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 404–436, Sofia, Bulgaria, April 2015.

[12] X. Fan and Q. Tang, "Making public key functional encryption function private, distributively," in *Proceedings of the Public-Key Cryptography-PKC 2018-21st IACR International Conference on Practice and Theory of Public-Key Cryptography*, pp. 218–244, Rio de Janeiro, Brazil, March 2018.

[13] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, pp. 40–49, Berkeley, CA, USA, October, 2013.

[14] Z. Brakerski and G. N. Rothblum, "Virtual black-box obfuscation for all circuits via generic graded encoding," in *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014 Lecture Notes in Computer Science*, Y. Lindell, Ed., Springer, San Diego, CA, USA, 2014.

[15] Y. Hu and H. Jia, "Cryptanalysis of GGH map," in *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques Lecture Notes in Computer Science*, M. Fischlin and J. Coron, Eds., Springer, Vienna, Austria, 2016.

[16] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology, Proc. CRYPTO' 84, LNCS 196*, G. R. Blakley and D. C. Chaum, Eds., pp. 47–53, Springer, Berlin, Germany, 1985.

[17] M. Bellare and G. Fuchsbauer, "Policy-based signatures," in *Proceedings of the Public-Key Cryptography-PKC 2014-17th International Conference on Practice and Theory in Public-Key Cryptography*, pp. 520–537, Buenos Aires, Argentina, March 2014.

[18] M. Backes, S. Meiser, and D. Schröder, "Delegatable functional signatures," in *Proceedings of the Public-Key Cryptography-PKC 2016-19th IACR International Conference on Practice and Theory in Public-Key Cryptography*, pp. 357–386, Taipei, Taiwan, March 2016.

[19] R. Johnson, D. Molnar, D. X. Song, and D. A. Wagner, "Homomorphic signature schemes," in *Proceedings of the Topics in Cryptology-CT-RSA 2002, the Cryptographer's Track at the RSA Conference, 2002*, pp. 244–262, San Jose, CA, USA, February 2002.

[20] D. M. Freeman, "Improved security for linearly homomorphic signatures: a generic framework," in *Proceedings of the Public Key Cryptography-PKC 2012-15th International Conference on Practice and Theory in Public Key Cryptography*, pp. 697–714, Darmstadt, Germany, May 2012.

[21] D. Catalano, D. Fiore, and B. Warinschi, "Homomorphic signatures with efficient verification for polynomial functions," in *Proceedings of the Advances in Cryptology-CRYPTO 2014-34th Annual Cryptology Conference*, pp. 371–389, Santa Barbara, CA, USA, August 2014.

[22] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the Advances in Cryptology-EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 457–473, Aarhus, Denmark, May 2005.

[23] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: definitions and challenges," in *Proceedings of the Theory of Cryptography-8th Theory of Cryptography Conference, TCC 2011*, pp. 253–273, Providence, RI, USA, March 2011.

[24] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich, "Reusable garbled circuits and succinct functional encryption," in *Proceedings of the Symposium on Theory of Computing Conference, STOC'13*, pp. 555–564, Palo Alto, CA, USA, June 2013.

[25] S. Goldwasser, S. D. Gordon, V. Goyal et al., "Multi-input functional encryption," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 578–602, Copenhagen, Denmark, May 2014.

[26] M. Abdalla, F. Bourse, A. D. Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," in *Proceedings of the Public-Key Cryptography-PKC 2015-18th IACR International Conference on Practice and Theory in Public-Key Cryptography*, pp. 733–751, Gaithersburg, MD, USA, April 2015.

[27] S. Agrawal, B. Libert, and D. Stehlé, "Fully secure functional encryption for inner products, from standard assumptions," in *Proceedings of the Advances in Cryptology-CRYPTO 2016-36th Annual International Cryptology Conference*, pp. 333–362, Santa Barbara, CA, USA, August 2016.

[28] F. Benhamouda, F. Bourse, and H. Lipmaa, "Cca-secure inner-product functional encryption from projective hash functions," in *Proceedings of the Public-Key Cryptography-PKC 2017-20th IACR International Conference on Practice and Theory in Public-Key Cryptography*, pp. 36–66, Amsterdam, The Netherlands, March 2017.

[29] T. Okamoto and K. Takashima, "Decentralized attribute-based signatures," in *Public-Key Cryptography-PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography Lecture Notes in Computer Science*, K. Kurosawa and G. Hanaoka, Eds., pp. 125–142, Springer, Nara, Japan, 2013.

[30] B. Liang and A. Mitrokotsa, "Decentralised functional signatures," *Mobile Networks and Applications*, vol. 24, no. 3, pp. 934–946, 2019.

[31] P. Datta, R. Dutta, and S. Mukhopadhyay, "Functional signcryption: notion, construction, and applications," in *Provable Security-9th International Conference, ProvSec 2015, Proceedings. Lecture Notes in Computer Science*, M. H. Au and A. Miyaji, Eds., Springer, Kanazawa, Japan, 2015.

[32] D. Pan, B. Liang, H. Li, and P. Ni, "Hierarchical functional signcryption: notion and construction," in *Provable Security-13th International Conference, ProvSec 2019 Lecture Notes in Computer Science*, R. Steinfeld and T. H. Yuen, Eds., pp. 167–185, Springer, Cairns, QLD, Australia, 2019.

[33] S. Li, B. Liang, and R. Xue, "Private functional signatures: definition and construction," in *Information Security and*

Privacy-23rd Australasian Conference, ACISP 2018 Lecture Notes in Computer Science, W. Susilo and G. Yang, Eds., Springer, Wollongong, NSW, Australia, 2018.

[34] M. Bellare, C. Namprempre, and G. Neven, "Unrestricted aggregate signatures," in *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007 Lecture Notes in Computer Science*, L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, Eds., Springer, Wroclaw, Poland, 2007.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6621760, 9 pages
https://doi.org/10.1155/2021/6621760

Research Article

Genetic Feature Fusion for Object Skeleton Detection

Yang Qiao, Yunjie Tian, Yue Liu, and Jianbin Jiao

School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

Correspondence should be addressed to Jianbin Jiao; jiaojb@ucas.ac.cn

Received 31 December 2020; Revised 28 January 2021; Accepted 24 February 2021; Published 9 March 2021

Academic Editor: Liguo Zhang

Object skeleton detection requires the convolutional neural networks to recognize objects and their parts in the cluttered background, overcome the image definition degradation brought by the pooling layers, and predict the location of skeleton pixels in different scale granularity. Most existing object skeleton detection methods take great efforts into the designing of side-output networks for multiscale feature fusion. Despite the great progress achieved by them, there are still many problems that hinder the development of object skeleton detection, such as the manually designed network is labor-intensive and the network initialization depends on models pretrained on large-scale datasets. To alleviate these issues, we propose a genetic NAS method to automatically search on a newly designed architecture search space for adaptive multiscale feature fusion. Furthermore, we introduce a symmetric encoder-decoder search space based on reversing the VGG network, in which the decoder can reuse the ImageNet pretrained model of VGG. The searched networks improve the performance of the state-of-the-art methods on commonly used skeleton detection benchmarks, which proves the efficacy of our method.

1. Introduction

Skeleton is an inherent visual descriptor of natural objects, which contains rich shape semantics of the objects. As compact topological representations of objects, the object skeletons boost the research of hand gesture recognition [1], human pose estimation [2], text detection [3], etc.

In the deep learning era, object skeleton detection methods are typically implemented by convolutional neural networks (CNNs) in an end-to-end manner and usually formulated as pixel-wise binary classification tasks. The essential difficulty of object skeleton detection can be summarized as how to simultaneously detect object skeletons with various scales since specific convolutional layers have limited receptive field size and lack of expression ability.

Based on the consensus that features from shallow layers contain low-level fine details while features from deep layers have rich high-level semantics, existing object detection works take great efforts on designing side-output networks for multiscale feature fusion. To name a few, holistically nested edge detection (HED) [4] pioneeringly detected object skeletons using abstracted multiscale representation extracted by the CNN, which are fused in a hierarchical parallel structure. Fusing scale-associated deep side-outputs (FSDS) [5] utilized scale-associated ground truth to supervise the network in a divided-and-conquer fashion, where deep side-output features with specific scale were fused gradually to predict skeleton maps in each layer. Side-output residual network (SRN) [6, 7] adopted side-output residual units to integrate the feature between adjacent stages from deep to shallow. Hi-Fi [8] introduced a bilateral feature integration mechanism to fuse features of different scales more densely. Despite great progress made by existing works on multiscale feature fusion, one limitation remains that the existing architectures are still hand-designed. The decoder network has no pretrained parameters. These disadvantages hinder the development of object skeleton detection to a certain extent.

In this paper, inspired by the remarkable success of neural architecture search (NAS) in many computer vision fields, such as image identification, object detection, and semantic segmentation, we propose a novel NAS-based object skeleton detection method, termed as genetic feature

fusion (GFF). Unlike the architectures of previous approaches, GFF introduces NAS to search the network architecture for adaptive feature fusion automatically. The follow-ups of HED [4] use the networks which are commonly used in classification, such as VGG [9], but it is difficult for these networks to have good adaptability on object skeleton detection. Guided by this view, we propose to expand the side-output architectures of the existing network and search the expanded search space to adaptively get the most suitable network architecture for object skeleton detection. In order to make full use of the pretrained parameters of these existing networks, the decoder is also reversed from the backbone (Figure 1). We utilize a genetic algorithm to search the expanded search space and get some novel networks which achieve state-of-the-art performance.

The contributions of this work are summarized as follows:

(i) We propose a novel NAS-based object skeleton detection method referred to as genetic feature fusion (GFF), which can make full use of pretrained parameters of existing networks both on encoder and decoder networks and adaptively fuse the multiscale features extracted in different convolutional layers

(ii) We introduce a search space expanded from the existing networks, which is designed to bridge the gap between the classification networks and object skeleton detection networks

(iii) We improve the state-of-the-art performance and demonstrate the effectiveness of our GFF on commonly used skeleton detection benchmarks

2. Related Work

Object skeleton detection is of great significance in the field of computer vision that skeletons reveal the inherent topological attributes of objects. In this section, we first review feature fusion architectures and then the object skeleton detection methods and finally summarize NAS technology.

2.1. Object Skeleton Detection. Convolutional neural networks (CNNs) play an absolutely dominant role in image-to-mask tasks. However, CNN itself suffers inherent contradictions, that is, good results require fine-grained scales from shallow features and abstract semantics from deep features. For alleviating this, researchers introduced and extensively studied feature fusion, which combines multiscale features to produce results. Object skeleton detection (symmetry detection) is an important image-to-mask task in computer vision. It helps to understand advanced features of objects in images, such as the topological property.

Early methods usually utilize geometric modelling, e.g., morphological image operation, to detect the skeleton of objects. Image segmentation procedures have to be performed as preprocessing when dealing with colour images. These methods always have good interpretability but poor performance.

Entering the era of deep learning, the performance of object skeleton detection has been improving greatly after the proposal of HED [4], which is a pioneer work to formulate the skeleton detection problem as a pixel-wise binary classification task. HED developed a parallel multiscale feature fusion with the help of a convolution neural network. Most methods proposed in the follow-up also treated object skeleton detection as a pixel-wise binary classification task and improved HED from the perspective of feature fusion. As shown in Figure 2, many popular works focused on the feature fusion to improve the performance.

Given scale-associated ground truth, fusing scale-associated deep side-output (FSDS) [5] learned the skeleton of objects using multiple network stages, which correspond to different scales. SRN [6] proposed a side-output residual network to expand the feature space to fit the errors between the output feature and the skeleton ground truth. Side-output residual units build short connections and send the deep coarse-grained features to the shallow fine-grained layer. In this way, features of different scales are merged to generate better skeleton maps. RSRN [7] utilized dense side-output residual units to make full use of richer multiscale features. Like FSDS, Hi-Fi [8] also used scale-associated supervision and built a hierarchical feature integration mechanism between stages of the network for the better fusion of features. LSN [10] increased the independence of different layer features using the linear span network and linear span units under the linear span view. Liu et al. [11] designed an orthogonal decomposition network to enrich feature diversity. PSG [12] uses Gaussian mixture models to partition skeleton branches for parametric skeleton generalization.

2.2. Neural Architecture Search. The recent rise of automated machine learning (AutoML) has caused extensive research because it automatically optimizes hyperparameters, which saves expensive expert knowledge. As an important part of AutoML, neural architecture search (NAS) is committed to automating the process of network architecture design and has found better network structures than manually designed.

NAS has three core components: search strategy, search space, and evaluation strategy. Four search strategies are commonly used, including reinforcement learning, evolutionary algorithms, Bayesian optimization, and one-shot methods. Early NAS is mainly formulated by reinforcement- and evolutionary-based methods but endures huge computational costs. The search process of NASNet [13], based on reinforcement learning, took 1800 GPU days, while the evolutionary-based AmoebaNet [14] took 3150 GPU days. The reason for such a huge computational overhead is that the sampled network structure must be independently trained in the evaluation phase. In order to reduce the computational cost, the one-shot method is proposed, which adopts the strategy of weight sharing to train the supernet, and the sampled networks inherit the weight of the supernet to avoid pretraining during evaluation. As a special family of weight sharing, gradient-based methods represented by DARTS [15] continue the discrete search space and achieve the purpose of searching

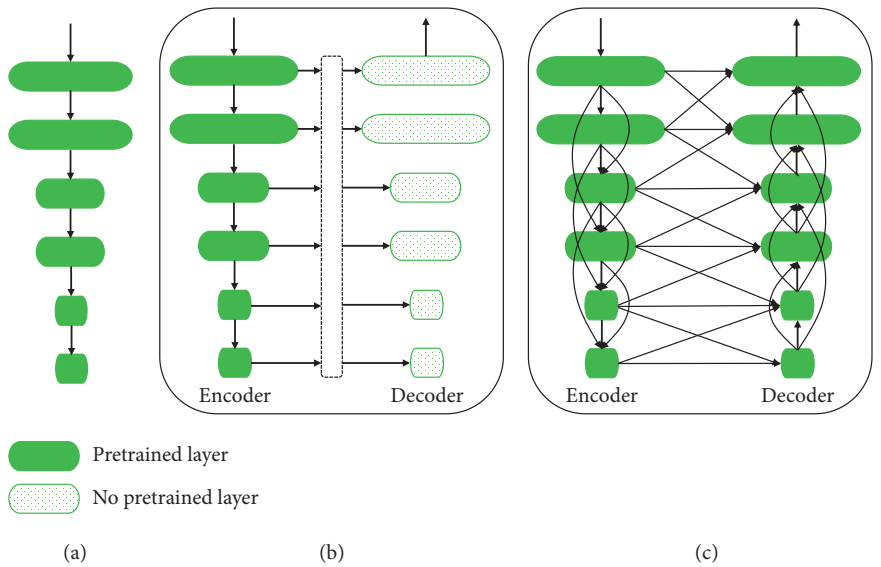


Figure 1: (a) A pretrained network migrated from classification; (b) the existing methods, of which the encoder directly utilized (a), and the decoder, after feature integration, used random initialized parameters; (c) our GFF with the expanded search space.

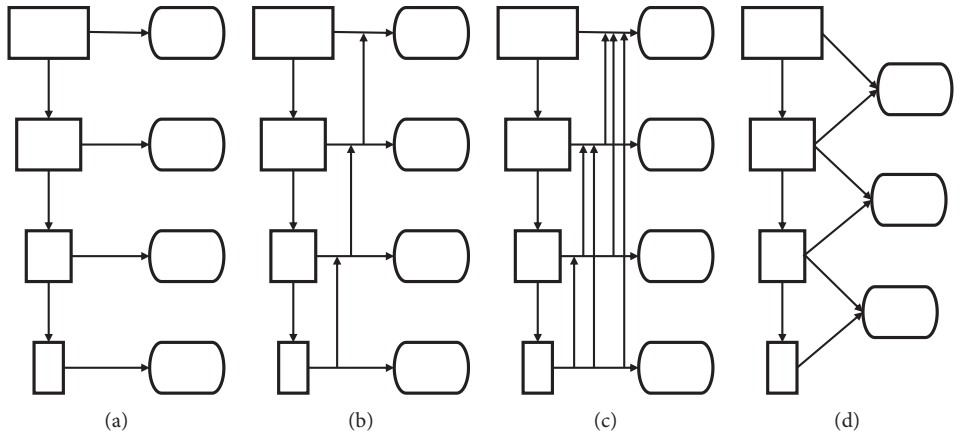


Figure 2: (a) Parallel feature integration architecture of HED [4]; (b) bottom-up feature integration which brings abstract semantics to detailed features [6]; (c) the linear span network proposed by Liu et al. [10] based on linear span theory to enrich the feature fusion; (d) hierarchical feature integration mechanism [8].

through the update of the gradient. Because of their fast realization of searching, usually within 1 GPU day, this kind of method has high hopes and is widely studied but still suffers difficulties to overcome instability.

Search space is a collection of neural networks, which refers to a space that contains all network architectures that could be searched. In order to simplify the search difficulty, the search space is usually limited to a cell space, called cell-based space [15–18]. Cell-based space had been extensively researched and achieved good performances, but it had been criticized for its low difficulty. Some works [19, 20] defined a search space based on the MobileNet [21] block (MBConv), benefiting from effective search and better candidate operations, usually achieving better results than cell-based methods. Thanks to the powerful MBConv, the recent EfficientNet [22], through the introduction of compound coefficient, defines the space about the three dimensions of network length, width, and image resolution and achieved convincing classification results through reinforcement learning.

Some recent methods have applied NAS not only on classification tasks but also to downstream tasks such as object detection and semantic segmentation and have also achieved state-of-the-art performances. In order to adaptively obtain the multiscale side-output fusion structure, NAS-FPN [23] defines a search space, in which the hierarchical feature fusion can be optimized. Under reinforcement learning, NAS-FPN achieves SOTA detection results. Auto-DeepLab [24] greatly improved the performance of segmentation through searching a well-designed search space, including the most popular hand-designed networks, e.g., U-net [25] and hourglass [26].

In this paper, we propose a genetic feature fusion (GFF) method. Based on VGG, we allow each convolutional layer

to accept inputs from multiple earlier convolutional layers, in this way, to achieve multiscale feature fusion, and to maximize the use of pretrained parameters. The network structure of GFF includes an encoder network and a decoder network. The inputs of each layer of the encoder can only come from the encoder, and the inputs of the decoder can come from both encoder and decoder. To reduce the complexity, we limit the number of inputs that can be used.

3. Genetic Feature Fusion

3.1. Revisiting Architecture Search Strategy. Enumerating all the architectures from the search space is the most straightforward strategy to search for the best network. Nevertheless, it has to deal with too enormous computational overhead to be acceptable, and heuristic methods become indispensable. The main heuristic methods of NAS are as follows: reinforcement learning, generic algorithm/evolutionary algorithm, and differentiable search. All these three strategies can achieve good results in a specific search space, so researchers turned to the heuristic search strategies. The basic framework is shown in Figure 3.

3.1.1. Reinforcement Learning. NASNet [16] first introduced reinforcement learning to the NAS community. Reinforcement learning-based methods treat architecture search as an agent learning process and predict good architectures by training the agent. Each network architecture can be represented by a character string, which is generated by an RNN controller. The strings are independently trained from scratch, and the correctness rate is used to train the agent. The agent then instructs the RNN controller to update its parameters so that better architectures can be probably generated next time. Neural architecture search can be performed by looping the above process. However, reinforcement learning-based NAS methods require lots of computational overhead. To predict good network architectures, they need to train a large number of generated architectures. For example, the search of NASNet [16] takes 1800 GPU days.

3.1.2. Genetic Algorithm. Genetic algorithms are inspired by the biological evolution process in nature, which retain superior individuals and discard inferior individuals through continuous iteration of populations that contain all the individuals. Genetic algorithms regard network architectures as individuals and network architecture search as the process of superior individuals gradually emerging from the evolution of the populations. Crossover, mutation, and selection are common operations of genetic algorithms. The crossover occurs between two individuals, each corresponding to its unique coded genes, and the two individuals exchange part of their genes with a predefined probability. The mutation is directed at a single individual, and it mutates with a certain probability to some genes corresponding to that one. Firstly, all the individuals in the population crossover one by one, then each individual is mutated once, and all the new individuals form the new population. After training all the individuals in the populations independently

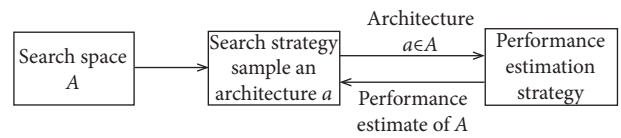


Figure 3: NAS framework consists of three core elements: search space, search strategy, and evaluation strategy.

from scratch, the selection of the superior individuals by certain criteria, such as accuracy rate, is called selection operation. Early genetic algorithm-based methods still require independent training of each network architecture from scratch, so these methods also require high computational costs, such as the search of AmoebaNet [14] which took 3150 GPU days. However, some of the recent methods have achieved state-of-the-art performance with fewer iterations and computation time [27, 28].

3.1.3. Differentiable Search. Reinforcement learning and genetic algorithm-based methods are slow because the evaluation phase takes lots of computational time; that is, every sampled network architecture is trained from scratch. To alleviate this issue, many methods have been proposed, such as using fewer parameters and smaller networks or searching on smaller proxy datasets, but these attempts simply bypass the computational overhead at the expense of a big gap, but the problem remains. Later, a weight-sharing mechanism is proposed to encode the entire search space into a supernetwork [15, 17, 29], in which each searched architecture corresponds to a subarchitecture. The modules in the supernetwork can be shared by all different subarchitectures so that the search procedure is largely accelerated. Differentiable architecture search strategy relaxes the weight-sharing supernetwork space to make it continuous by adding a weight for each candidate operation. DARTS [15], one of the most popular differentiable search algorithms, defined a cell-based search space, which initializes a weight $1/C$ of C candidate operations of each node. Then, these weights can be updated in a continuous space by gradient optimization. The procedure of updating the weights corresponding to these candidate operations is the procedure of searching architecture. At the end of searching, operations with higher weights are retained, and operations with lower weights are discarded, that is, a discretization process is performed.

3.2. Architecture Space for Feature Fusion. The overall network structure has two parts: encoder and decoder, both of which are VGG networks where the decoder network is equivalent to the reversed VGG network structure, except that the pooling layers in the decoder are all replaced by upsampling operations (Figure 4). Including the pooling layers, VGG-16 has 18 layers in all, which are labelled 1–18 sequentially. It is worth to note that the deeper the layer is in the encoder, the larger the number is. For example, the 7th layer in the encoder is after the 6th layer, so the deeper layer's number is larger, while the deeper the layer is in the decoder, the smaller the number is. For example, the 6th decoder layer

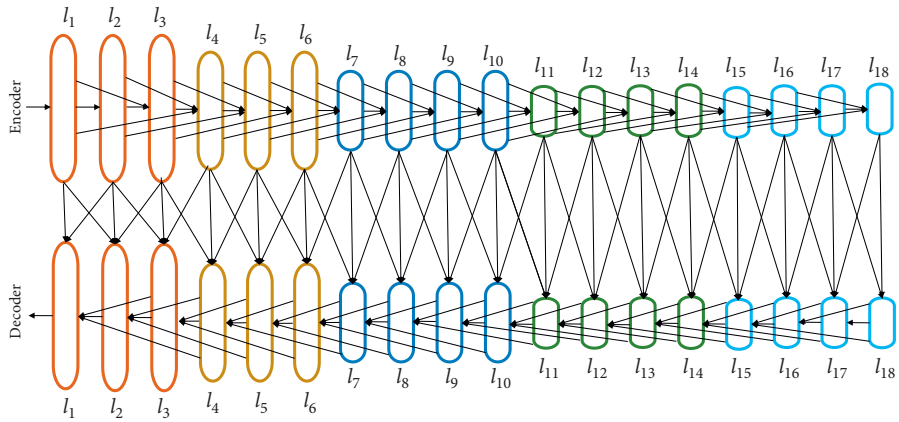


Figure 4: Diagram of the expanded search space based on the VGG [9] network.

is after the 7th decoder layer, that is, the 6th layer is deeper and closer to the first decoder layer.

The search scope is the input of each convolution layer in the encoder and decoder, that is, where the input of each layer comes from. We make a constrain that the input of each convolution layer in the encoder can only come from the encoder, and each convolution layer in the decoder can come from both decoder and the encoder. In order to simplify the space, the range of the candidate input of each convolution layer in the encoder is constrained to three, that is to say, each convolution layer can only accept the output of the three layers before as its input. For example, the 5th layer can only accept the outputs of the second, third, and/or fourth layers. As the decoder can accept the input from both the encoder and the decoder itself, the range of candidate inputs for each layer (convolution or pooling) in the decoder is specified to be 6, where the upper range of the decoder layer is 3, and the encoder layer is 3. The examples of acceptable inputs for each decoder layer are as follows: the 6th layer in the decoder network can accept the 9th, 8th, and 7th decoder layers and the 5th, 6th, and 7th encoder layers.

3.3. Genetic Architecture Search Algorithm. The search algorithm is similar to AdaLSN [27]. As described above, each layer in the network is numbered differently for the indexing purpose. The input of each layer can be represented by a simple list, called the genome, in which each element, called a gene, indicates whether the candidate input layer is selected as the input so that a network individual in the search space can be uniquely encoded as a list set, called a genotype. We specify that each layer accepts the first three layers as inputs and forces the previous layer to be retained to ensure that the entire VGG network is retained. So, except for the marginal layer, there are two genes in each genome on the encoder and five genes in each genome on the decoder.

We randomly initialize 24 individuals to form the initial population. Each individual is trained for 1000 steps from scratch, which is enough for a network to be trained well since most of the parameters are retained. Then, the evaluation process is performed on the validation set that contains 50 images separated from the training set. The eight individuals with the lowest losses are selected as the dominant ones.

Crossover process occurs on the eight selected dominant individuals, paired in pairs, and candidate inputs of each layer, a genome, are exchanged with probability so that 8 new individuals are generated. Each gene of each new individual is mutated with probability p_m, resulting in 8 new individuals forming a new population. Figure 5 shows the overall process. The preceding process is executed for 50 times, and then the final architecture is obtained.

The final network structure will be followed by a convolution to reduce the number of channels to 1, crop the gray image to the size of the original image, and then supervise its distance from the ground truth through the commonly used binary cross-entropy.

4. Experiments

4.1. Experimental Setting

4.1.1. Datasets. We trained and tested our method on 3 datasets which are commonly used for object skeleton detection: SK-LARGE [30], SK506 [5], and WH-SYMMAX [31]. SK-LARGE is sampled from the MS-COCO dataset [32], which contains 746/745 images for training and testing, with 16 classes of objects included. SK506, also called SK-SMALL, is the old version of SK-LARGE, which contains 300 training images and 206 test images in 16 different objects. WH-SYMMAX contains 200/100 training and test images all about Weizmann Horse. All training images and ground truths are augmented with scaling (0.8x, 1.0x, and 1.2x), rotating (0°, 90°, 180°, and 270°), and flipping (right to left and left to right), as well as resolution normalization [33].

4.1.2. Implementation. The search process is executed on NVIDIA TITAN RTX GPUs (24 GB of memory). VGG [9] is used as the backbone network for evolution. The crossover rate p_e and mutation rate p_m are both 0.2 and 50 generations in total. One final searched architecture is shown in Figure 6.

The final architecture was trained and tested on single NVIDIA GTX 2080Ti (12 GB of memory). We used the

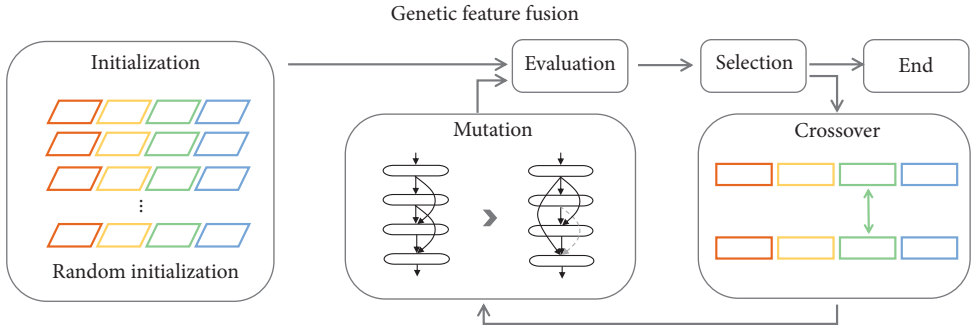


Figure 5: Framework of our GFF. The initial population is evaluated, which adopts the random initialization, and dominant individuals perform crossover and mutation to form a new population, and then the above process is iterated.

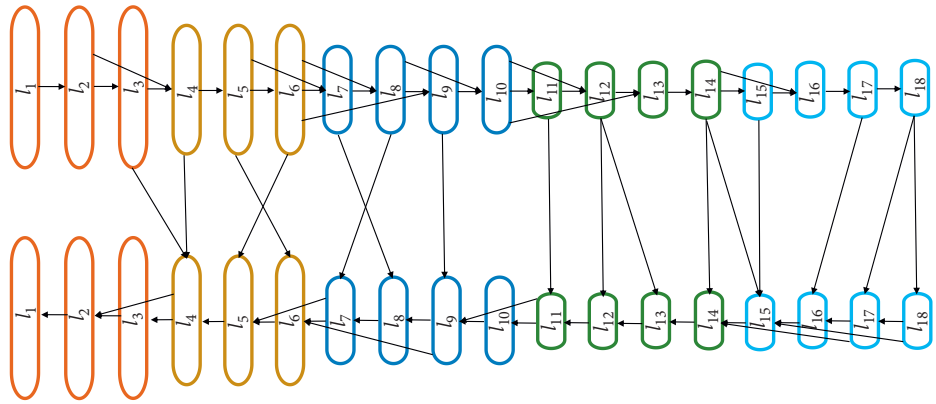


Figure 6: Diagram of one searched architecture.

Adam optimizer [34] with the initial learning rate of $1e{-}7$, weight decay of 0.0002, and betas of (0.9, 0.999). During training, if batch size is 1, the network parameter would be updated using the added gradient calculated every 10 iterations of forwarding propagation. The learning rate is fixed during the first 50,000 iterations and decreased to one-tenth of the original. The training process would be stopped in iteration 60,000. The parameters of the searched architecture are about 32.1 M, with flops of 72.6 G when inputting a 224×224 image, and the training process took 9.5 hours.

The setting in [35] is used as the evaluation metrics.

4.2. Performance and Comparison. Our GFF outperforms the state of the art of object skeleton detection in several commonly used datasets, and the results are shown in Table 1. VGG [9] with pretrained parameters is used for our proposed method, and the searched architecture is retrained on each dataset.

The skeleton results are shown and compared with other methods in Figure 7. One can see the superiority of our GFF, which can extract skeleton maps of different granularities with better continuity.

On the SK-LARGE [30] dataset, our GFF achieves the performance of F-score 73.6%, which is the highest object skeleton detection performance compared to the optimal method DeepFlux [37], which achieves 73.2%. We also outperform Hi-Fi [8], which utilizes the additional scale-associated ground truth with a margin of 1.2%. In another convincing dataset SK506 [5], GFF also embodies superiority. GFF achieves the F-score of 72.3%, which is higher than Hi-Fi [8] and DeepFlux [37] with margins of 4.2% and 2.3% separately.

4.3. Ablation Study. We verify the effectiveness of our method through comparative experiments which perform search process randomly. We randomly reserve each candidate operation for each layer and execute 4 times independently to get 4 architectures. Each architecture is retrained on SK-LARGE for the same number of steps, and the results are shown in Table 2. One can find that the results of these 4 architectures range from 70.9% to 72.7%, which is lower than 73.6% achieved by the architecture we searched.

Figure 8 shows the adaptiveness of the GFF during the search process. Figure 8(a) shows the change in the loss of top1–4 in different generations. The values of loss are obviously decreasing, which indicates that better architectures appear in the population as it evolves. Figure 8(b) shows the change in the number of candidate input connections reserved on the encoder and the decoder in different generations. In general, the number of candidate connections is decreasing, which indicates that our method is adaptively searching for scale-aware architectures. Figure 8(c) shows

Table 1: Performance comparison of the state-of-the-art approaches on commonly used skeleton\symmetry detection datasets.

Methods	Datasets		
	SK-LARGE [30]	SK-SMALL [5]	WH-SYMMAX [31]
MIL [35]	0.353	0.392	0.365
HED [4]	0.497	0.541	0.732
RCF [36]	0.626	0.613	0.751
FSDS [5]	0.633	0.623	0.769
SRN [6]	0.658	0.632	0.443
LSN [10]	0.668	0.633	0.797
Hi-Fi [8]	0.724	0.681	0.805
DeepFlux [37]	0.732	0.695	0.840
GFF (ours)	0.736	0.723	0.837

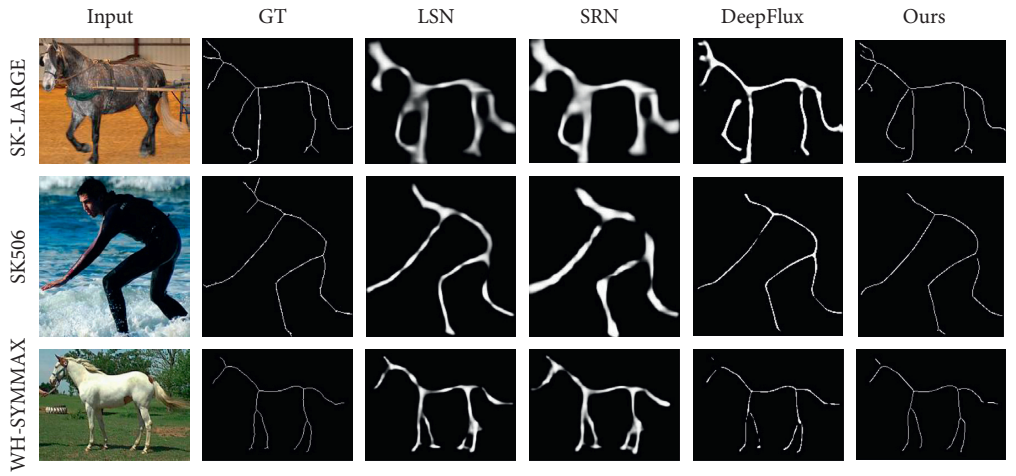


Figure 7: Skeleton detection examples by state-of-the-art approaches on SK-LARGE [30], SK-SMALL [5], and WH-SYMMAX [31].

Table 2: Performance of random search.

Archi1	Archi2	Arch31	Archi4
0.714	0.723	0.709	0.727

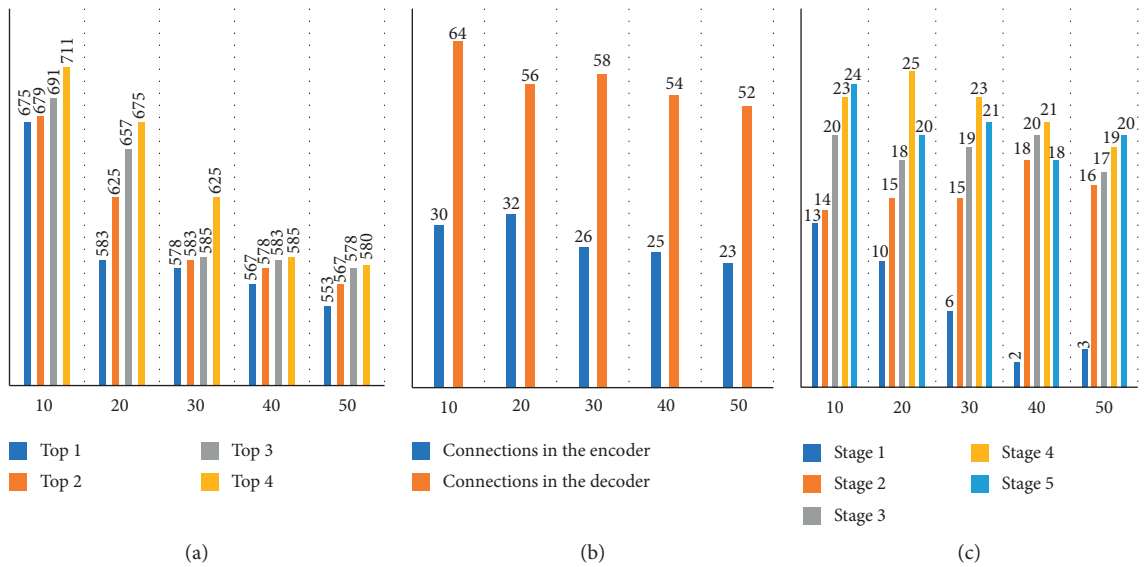


Figure 8: The horizontal coordinate of all figures indicates the generations. (a) Changes of loss of top1–4 architectures. (b) Changes in the number of connections reserved for the encoder and decoder. (c) Changes in the number of connections reserved for stages 1–5 in the framework.

Table 3: Other models.

VGG	FHN [38]
0.480	0.675

the number of candidate input connections reserved in different stages. One can find that the connections of all stages except stage 2 are decreasing because the features of stage 2 are well-balanced, fine-grained, and coarse-grained features, but other stages reserve fewer and fewer candidate input connections, which makes sense of our method's search process.

To demonstrate the superiority of our GFF, in Table 3, we test VGG which achieves the F-score of 48.0% that is 25.6% lower than ours. FHN [38], with similar motivation, is 6.1% lower than our GFF.

5. Conclusion

Object skeleton detection has aroused widespread research interest in the field of computer vision, helping to understand the attributes of shape and topology of objects in natural images. In this paper, we proposed genetic feature fusion (GFF), which expanded the existing classification networks to form a network space set. Each network, including an encoder and decoder, in this space could make full use of pretrained parameters. Driven by neural architecture search (NAS), GFF automatically integrates features of different scales and searches the scale-aware networks for object skeleton detection in the expanded space. The significant higher performance in commonly used datasets demonstrated the superiority of GFF.

Data Availability

The dataset used in our paper can be made available at https://kaizhao.net/sk-large https://openaccess.thecvf.com/content_cvpr_2016/html/Shen_Object_Skeleton_Extraction_CVPR_2016_paper.html https://dl.acm.org/doi/10.1016/j.patcog.2015.10.015.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) (Grant nos. 61836012 and 61771447) and Strategic Priority Research Programme of Chinese Academy of Sciences (Grant no. XDA27010303).

References

[1] C. L. Teo, C. Fermüller, and Y. Aloimonos, "Detection and segmentation of 2d curved reflection symmetric structures," in *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pp. 1644–1652, Santiago, Chile, December 2015.

[2] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the*, pp. 4724–4732proceeding 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, June 2016.

[3] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proceedings of the*, pp. 2558–2567proceeding 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, June 2015.

[4] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pp. 1395–1403, Santiago, Chile, December 2015.

[5] W. Shen, K. Zhao, Y. Jiang, Y. Wang, Z. Zhang, and X. Bai, "Object skeleton extraction in natural images by fusing scale-associated deep side outputs," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 222–230, Las Vegas, NV, USA, June 2016.

[6] W. Ke, J. Chen, J. Jiao, G. Zhao, and Q. Ye, "SRN: side-output residual network for object symmetry detection in the wild," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 302–310, Honolulu, HI, USA, July 2017.

[7] C. Liu, W. Ke, J. Jiao, and Q. Ye, "RSRN: rich side-output residual network for medial axis detection," in *Proceedingsof the 2017 IEEE International Conference on Computer Vision Workshops*, pp. 1739–1743, Venice, Italy, October 2017.

[8] K. Zhao, W. Shen, S. Gao, D. Li, and M. Cheng, "Hi-fi: Hierarchical feature integration for skeleton detection," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence {IJCAI-18}*, pp. 1191–1197, Stockholm, Sweden, July 2018.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, http://arxiv.org/abs/1409.1556v6.

[10] C. Liu, W. Ke, F. Qin, and Q. Ye, "Linear span network for object skeleton detection," in *Proceedings of the 2008 15th European Conference on Computer Vision*, pp. 133–148, Munich, Germany, September 2018.

[11] C. Liu, F. Wan, X. Zhang, Y. Yao, W. Ke, and Q. Ye, "Orthogonal decomposition network for pixel-wise binary classification," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6064–6073, Long Beach, CA, USA, June 2019.

[12] C. Liu, D. Luo, Y. Zhang, W. Ke, F. Wan, and Q. Ye, "Parametric skeleton generation via Gaussian mixture models," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1167–1171, Long Beach, CA, USA, June 2019.

[13] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2017.

[14] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the Thirty-Third AAAI Conference on artificial intelligence, AAAI 2019*, Honolulu, HI, USA, January 2019.

[15] H. Liu, K. Simonyan, and Y. Yang, "DARTS: differentiable architecture search," 2019, http://arxiv.org/abs/1806.09055v2.

[16] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8697–8710, Salt Lake City, UT, USA, June 2018.

[17] Y. Xu, L. Xie, X. Zhang et al., "PC-DARTS: partial channel connections for memory-efficient differentiable architecture search," 2020, http://arxiv.org/abs/1907.05737.

[18] Y. Tian, C. Liu, L. Xie, J. Jiao, and Q. Ye, "Discretization-aware architecture search," 2020, https://arxiv.org/abs/2007.03154.

[19] M. Tan, B. Chen, R. Pang et al., "Mnasnet: platform-aware neural architecture search for mobile," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, June 2019.

[20] J. Fang, Y. Sun, Q. Zhang, Y. Li, W. Liu, and X. Wang, "Densely connected search space for more flexible neural architecture search," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, USA, June 2020.

[21] A. G. Howard, M. Zhu, B. Chen et al., "Efficient convolutional neural networks for mobile vision applications," 2017, https://arxiv.org/abs/1704.04861v1.

[22] T. Mingxing and Q. V. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," in *Proceedings of the 2019 36th International Conference on Machine Learning*, Long Beach, CA, USA, June 2019.

[23] G. Ghiasi, T. Lin, and Q. V. Le, "NAS-FPN: learning scalable feature pyramid architecture for object detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7036–7045, Long Beach, CA, USA, June 2019.

[24] C. Liu, L. Chen, F. Schroff et al., "Auto-deeplab: hierarchical neural architecture search for semantic image segmentation," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 82–92, Long Beach, CA, USA, June 2019.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the 2015 18th International Conference*, Munich, Germany, October 2015.

[26] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proceedings of the Computer Vision-ECCV 2016 14th European Conference*, Amsterdam, The Netherlands, October 2016.

[27] C. Liu, Y. Tian, J. Jiao, and Q. Ye, "Adaptive linear span network for object skeleton detection," 2020, https://arxiv.org/abs/2011.03972v1.

[28] Y. Chen, Q. Zhang, C. Huang, L. Mu, G. Meng, and X. Wang, "Reinforced evolutionary neural architecture search," 2018, https://arxiv.org/abs/1808.00193.

[29] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Stockholm, Sweden, July 2018.

[30] W. Shen, K. Zhao, Y. Jiang, Y. Wang, X. Bai, and A. L. Yuille, "Deepskeleton: learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images," *IEEE Transactions on Image Processing*, vol. 26, no. 11, 2017.

[31] W. Shen, X. Bai, Z. Hu, and Z. Zhang, "Multiple instance subspace learning via partial random projection tree for local reflection symmetry in natural images," *Pattern Recognition*, vol. 52, pp. 306–316, 2016.

[32] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 664–676, 2017.

[33] W. Xu, G. Parmar, and Z. Tu, "Geometry-aware end-to-end skeleton detection," 2019.

[34] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[35] S. Tsogkas and I. Kokkinos, "Learning-based symmetry detection in natural images," in *Proceedings of the 2012 European Conference on Computer Vision*, Florence, Italy, October 2012.

[36] Y. Liu, M. Cheng, X. Hu et al., "Richer convolutional features for edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1939–1946, 2019.

[37] Y. Wang, Y. Xu, S. Tsogkas, X. Bai, S. J. Dickinson, and K. Siddiqi, "Deepflux for skeletons in the wild," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5287–5296, Long Beach, CA, USA, June 2019.

[38] N. Jiang, Y. Zhang, D. Luo, C. Liu, Y. Zhou, and Z. Han, "Feature hourglass network for skeleton detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Long Beach, CA, USA, June 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6633250, 11 pages
https://doi.org/10.1155/2021/6633250

Research Article

ETCC: Encrypted Two-Label Classification Using CNN

Yan Li [1] **and Yifei Lu** [1,2]

[1]*Nanjing University of Science and Technology, Nanjing 210094, China*
[2]*State Key Lab of Mathematical Engineering and Advanced Computing, Wuxi 214215, China*

Correspondence should be addressed to Yifei Lu; luyifei@njust.edu.cn

Received 1 January 2021; Revised 26 January 2021; Accepted 17 February 2021; Published 8 March 2021

Academic Editor: Liguo Zhang

Due to the increasing variety of encryption protocols and services in the network, the characteristics of the application are very different under different protocols. However, there are very few existing studies on encrypted application classification considering the type of encryption protocols. In order to achieve the refined classification of encrypted applications, this paper proposes an Encrypted Two-Label Classification using CNN (ETCC) method, which can identify both the protocols and the applications. ETCC is a two-stage two-label classification method. The first stage classifies the protocol used for encrypted traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. Experimental results show that the ETCC achieves 97.65% accuracy on a public dataset (CICDarknet2020).

1. Introduction

According to the forecast of Cisco's Annual Internet Report [1], by 2023, the total number of global Internet users will increase from 3.9 billion in 2018 (51% of the global population) to 5.3 billion (66% of the global population), and the number of devices connected to IP networks will reach 29.3 billion, more than three times the global population. As more users and devices connect to the network, the applications will become more and more diversified, and Internet communication methods will become more and more complex, which also make network management more complicated [2]. But if we can identify the application type of network traffic, we can improve the level of network management. For example, many applications occupy a large amount of network bandwidth, causing other applications to operate abnormally. If Internet Service Providers (ISPs) can provide different levels of service quality according to different types of applications, the unfair use of network resources can be solved [3], and the user's Internet experience will also be better.

On the other hand, in order to meet the needs of users for security and privacy, various encryption technologies are widely used in network communications [4]. Security Socket Layer (SSL), Virtual Private Network (VPN), Secure Shell (SSH), and The Onion Router (Tor) are currently the most common encryption methods [5]. But encryption not only protects users' privacy but also poses other threats to users. Encryption technologies can help hackers hide their malicious behavior. Network managers need to be able to identify encrypted traffic in a timely manner, so as to quickly and accurately locate attacks on the network, cut off the transmission path, and reduce the harm of malicious behavior to users. Encryption also causes trouble to the IT team. The payload will change after the traffic is encrypted. This change brings additional challenges to the accurate identification of encrypted network protocols and encrypted network applications, resulting in the complexity and difficulty of the traffic analysis and network management [6]. Moreover, even if it can be classified accurately, it is difficult to guarantee real-time performance.

Encryption invalidates many early traffic classification methods, such as port-based classification, entropy-based classification, payload-based classification, and pattern matching-based classification. This is because the port, entropy, payload, and header of network traffic will change

with encryption [6]. In recent years, machine learning methods have been the most commonly used method for classifying encrypted traffic. This is because encryption is usually only for the payload, and the machine learning method only care about statistical features, not the value of the payload. Hence, machine learning methods are less affected by encryption. This makes machine learning based methods more accurate than other methods.

Most encrypted application classification methods are based on single label. In other words, they directly use the classifier to determine the application type of network traffic. But under different encryption protocols, the characteristics of the application are also different. The encryption protocol mainly has two steps, the initialization of the connection and the transmission of encrypted data. The initialization of the connection is divided into initial handshake, identity verification, and shared key establishment. Because the encryption principles of different encryption protocols are different, these steps are very different, which leads to different representation of the final encrypted traffic [5]. Therefore, if we can classify encrypted applications on the basis of known encryption protocols, we can get more accurate results than single-label classification.

In this paper, we propose an Encrypted Two-Label Classification method, referred to as ETCC, to improve the accuracy of encrypted application classification. ETCC is a two-stage two-label classification method. The two labels are encryption protocol and application. The first stage classifies the protocol used for encrypted traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. The contributions of this paper are summarized as follows:

(1) We propose a two-stage two-label scheme called ETCC, which carries out refined application classification according to the encryption protocol used

(2) In the second stage of application classification, encrypted traffic can select the corresponding classifier according to the protocol type, instead of uniformly using the same classifier

(3) Our scheme can identify both the protocol and the application, which can meet various needs

The rest of this paper is organized as follows. Section 2 introduces some encryption traffic classification methods and some multilabel classification methods. In Section 3, a scheme is proposed to achieve refined applications classification. And some experiments and evaluations are presented in Section 4. Finally, Section 5 concludes our work and proposes some future works.

2. Related Work

In this section, we introduce some methods for classifying encrypted traffic and methods for multilabel classification. These works give bright inspiration for our research.

In the early research, the commonly used methods include port-based classification, entropy-based classification, payload-based classification, and pattern matching-based classification. In the early days of Internet development, every application had a fixed port number assigned by IANA [7]. Therefore, we only need to check the IANA TCP/UDP list to know the type of application. However, with the emergence of technologies such as port confusion and network address translation (NAT), port-based methods have become no longer feasible. Entropy-based methods classify encrypted traffic by extracting geometric features between traffic. Casino et al. [8] propose a method to distinguish encrypted and nonencrypted traffic based on the entropy value. They only analyze a random subset, not the complete network traffic, to ensure real-time performance. The payload-based method can no longer analyze the contents of the package and cannot be used anymore [9]. The method based on pattern matching judges whether it is encrypted traffic and encryption protocol type by checking the header format but cannot further judge the application type. In summary, we need more advanced methods to achieve encrypted traffic classification task.

The most commonly used method is based on machine learning. The differences of these methods are reflected in feature extraction, model selection, and parameter setting. Liu et al. [10] only consider first N packets in a sliding window, which not only reduces the dimension of encrypted traffic characteristics but also reduces the number of data packets in each flow. Similarly, Hasan et al. [11] analyze the first 64 packets to identify Android applications. Finally, they state that most Android applications can be identified through the TCP/IP header. Shen et al. [12] combine the certificated packet length and the first application data size as a unique fingerprint for a given application and then use the second-order Markov chain to classify encryption applications. Cui et al. [13] propose the SPCaps model, which uses capsule neural networks (CapsNet) to learn the spatial features of encrypted traffic. The advantage of this model is that it simultaneously learns the position of the feature in the package and the order between the packages. Ly Vu et al. [14] used time series as an entry point to classify encrypted traffic. Their method is divided into two steps. The first step is to extract behavior patterns based on the time series of packets. The second step is to classify according to the correlation between time series samples. Zeng et al. [15] think more comprehensively. Their scheme not only analyzes spatial features but also analyzes temporal features and coding features. However, these works still ignore the suddenness of network traffic and cannot capture complex nonlinear features. The framework proposed in [16] leverages multifractal feature extraction technology, which can capture the self-similarity of network traffic structure in a wide time range. Because it is always difficult to consider comprehensively when extracting features, Wang et al. [17] took a different approach and directly converted the flow into a picture and put it into the model for classification. Lotfollahi et al. [18] employ CNN and SAE to classify encrypted traffic, respectively. There is no need for an expert to extract features and provide reference for many later studies.

The classification in general scenarios was introduced earlier, but, for specific scenarios, using specific methods can be more efficient. Shen et al. [19] introduce the traffic

classification in Ethereum. Because these flows are all generated on the same platform, it will be more difficult to distinguish. To this end, they study where the existing methods are easy to misclassify and extract features from three aspects: packet length, packet burst, and time series. In order to evaluate quality of experience (QoE) and bring better services to users, Orsolic et al. [20] propose a system for YouTube videos called YouQ. They collect YouTube videos and evaluate the QoE of the videos based on the traffic characteristics of each video session. Similarly, Tarun Mangla et al. [21] evaluated the QoS of encrypted HTTP-based adaptive streaming (HAS) sessions. Anderson et al. [22] analyze TLS encrypted sessions in commercial malware sandboxes and two enterprise networks. They claim that the choice of features has a great impact on performance. In order to monitor and detect specific users, Pierre-Olivier Brissaud et al. [23] propose a scheme for monitoring HTTP/2 communication based on the TLS protocol. This scheme is designed to detect whether the user has performed certain specified operations. The QUIC (Quick UDP Internet Connection) protocol is a new default encrypted Internet communication protocol that provides many improvements to speed up HTTP communication while making it more secure. However, since it is a new type of protocol, the amount of data available is very small. Rezaei et al. [24] propose a semisupervised learning based method that first trains the model with a large amount of unlabeled data and then retrains the model with a small amount of labeled data. For network traffic classification, it greatly reduces the amount of labeled data required.

The studies on multilabel classification are very few. There are two common ways to deal with multilabel classification. Convert the multilabel classification problem into several single-label classification methods, or integrate multilabels into a single label. Grigorios Tsoumakas et al. [25] give a detailed introduction to multilabel classification and compare several classification methods, which provide a lot of guidance for our research. Tien Thanh Nguyen et al. [26] propose a Bayes-based method that not only considers the relationship between labels and features but also considers the relationship between label pairs. Jesse Read et al. [27] constructed a multilabel Hoeffding tree with classifiers at the leaves. Moreover, they create a new set of benchmarks in predictive performance and time complexity. Darshin Kalpesh Shah et al. [28] use RNN and LSTM to classify multilabel text. The performance is significantly better than Logistic Regression and ExtraTrees. Ou Guangjin et al. [29] present a graph convolution networks based multilabel zero-shot learning model to recognize novel categories. Most of the multilabel classification is aimed at the problem of category independence. However, Nadia Ghamrawi et al. [30] study the problem of high label dependence. Jesse Read et al. [31] also study the high dependency between labels. They use a chaining method to model the label relationship. Pengcheng Yang et al. [32] regard the multilabel classification task as a sequence generation problem and used the sequence generation model for classification. Experiments show that this method can effectively capture the correlation between labels. These works help us a lot. Similarly, a two-stage two-label method is proposed in our paper, in which the protocols are classified in the first stage, and then applications are classified in the second stage. The biggest difference between our method and other multilabel classification methods is that our method will select the corresponding classifier for the second stage classification based on the results of the first stage. We achieve refined classification and two-label classification can meet various needs.

3. Methodology

In this section, we propose a two-stage, two-label scheme to classify encrypted applications, called ETCC. Our scheme consists of three modules: preprocessing, first label and second label module. They are used to preprocess data, classify protocols, and classify applications, respectively. Figure 1 presents the details.

3.1. Preprocessing Module. This module is used to process raw data and convert them into a format suitable for the input of the classifier.

First, we collect some encrypted traffic and label them with protocols and applications.

Second, we select and extract some features. A flow is a collection of packets with the same IP five-tuple {Source IP, Destination IP, Source Port, Destination Port and Protocol}. Because the packets of the same flow are usually the same encryption protocol and application, we process data in units of flows. We use spatial features and temporal features to distinguish encrypted traffic, because these two features are not easily affected by encryption. Spatial features are related to quantity and size. Temporal features are features associated with time series. The specific features are shown in Table 1.

Third, we use the Sequential Floating Forward Selection (SFFS) algorithm [33] to select the most suitable features. We finally selected 41 features about Port, Protocol, Flow Duration, Length of Packet, Flow Bytes/s, Packets/s, Flow IAT, Forward IAT, Backward IAT, Flag Count, and Active Time. Detailed features are shown in Table 1. Through these simplified features, we can get a classifier with better generalization and faster speed.

Finally, we apply Min-Max Scaling [34] to normalize feature to meet the input requirements of supervised classifier and speed up model training. The formula of Min-Max Scaling is shown in

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}, \tag{1}$$

where $X_{\max}$ is the maximum value of the sample data, $X_{\min}$ is the minimum value of the sample data,X is the current sample value, and X_{norm} is the normalized value of the current sample.

After this, the feature values are all mapped to the interval [0,1] and fed into the first label module.

3.2. First Label Module. We leverage this module to classify various encryption protocols into m categories. At first, we choose CNN and LSTM classifiers and test their

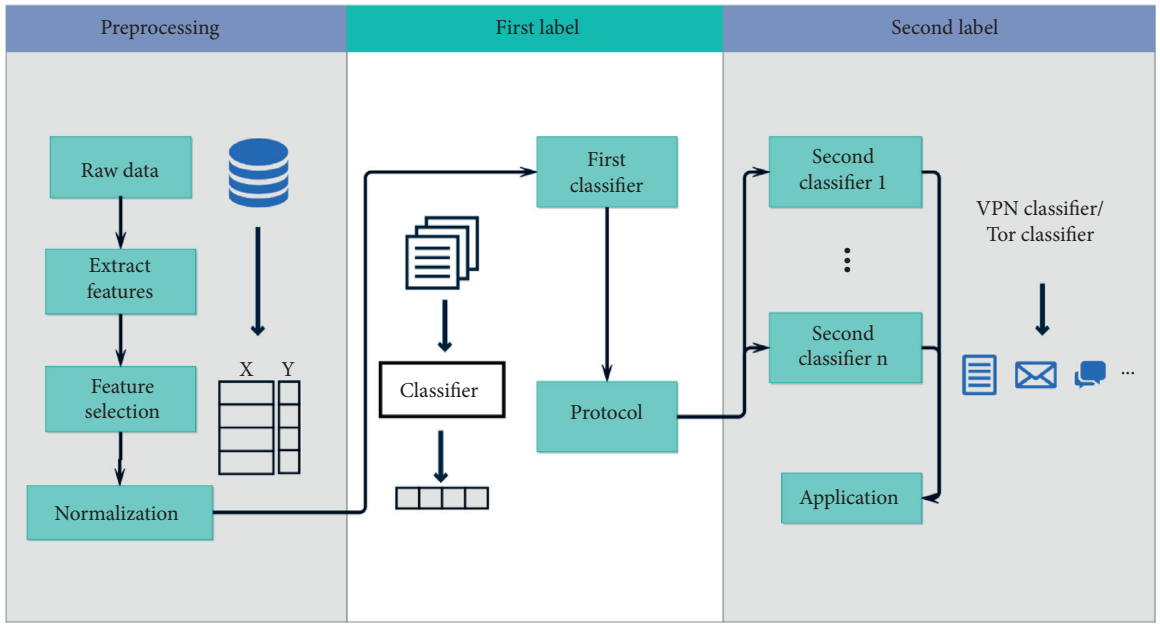


Figure 1: System framework.

Table 1: Features details.

Feature	Description
IP	{Source IP, destination IP}
Port	{Source port, destination port}
Protocol	The protocol of the flow
Flow duration	The duration of the flow
Packet	Total packets in the {forward, backward} direction
Length of packet	{Total, max, min, mean, std} size of packet in {forward, backward} direction
Flow packet length	{Max, min, mean, std, variance} length of a flow
Flow bytes/s	# of bytes transferred per second
Flow packets/s	# of packets transferred per second
Packets/s	# of {forward, backward} packets per second
Flow IAT	{Max, min, mean, std} time between two flows
Forward IAT	{Total, max, min, mean, std} time between two packets sent in the forward direction
Backward IAT	{Total, max, min, mean, std} time between two packets sent in the backward direction
Flags	# of times the {PSH, URG} flag was set in packets travelling in the {forward, backward} direction (0 for UDP)
Flag count	# of packets with {FIN, SYN, RST, PSH, ACK, URG, CWE, ECE}
Header length	Total bytes used for headers in the {forward, backward} direction
Ratio	{Down, up} ratio
Average packet size	Average size of packet
Segment size avg	Average size observed in the {flow, forward, backward} direction
Bytes/Bulk avg	Average number of bytes bulk rate in the {forward, backward} direction
Packet/Bulk avg	Average number of packets bulk rate in the {forward, backward} direction
Bulk rate avg	Average number of bulk rate in the {forward, backward} direction
Subflow packets	The average number of packets in a subflow in the {forward, backward} direction
Subflow bytes	The average number of bytes in a subflow in the {forward, backward} direction
Init win bytes	# of bytes sent in initial window in the {forward, backward} direction
Forward Act data pkts	# of packets with at least 1 byte of TCP data payload in the forward direction
Forward seg size min	Minimum segment size observed in the forward direction
Active time	{Mean, max, min, std} time a flow was active before becoming idle
Idle time	{Mean, max, min, std} time a flow was idle before becoming active

performance, respectively. In the end, we apply CNN, which performs better. The reason for applying CNN is addressed in Section 4.3.

Figure 2 depicts the architecture for CNN. It contains of convolution, pooling, flatten, and dense layers. The convolution layer is used to extract different features of the

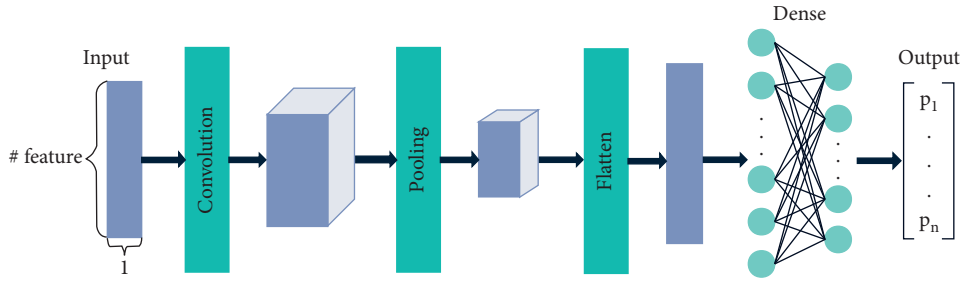


Figure 2: CNN architecture.

input. However, if several convolution layers are used continuously, the amount of calculation will become very large, and the pooling layer can effectively reduce the amount of calculation through downsampling. Next, the flatten layer will convert the convolved data to one-dimensional and facilitate connection to the dense layer. The dense layer combines all local features into global features at the end to get the classification results.

Figure 3 depicts the architecture for LSTM. The input layer and output layer of LSTM are similar to CNN, but the difference lies in the intermediate calculation process. LSTM cells can learn two pieces of information: new input information and previous memory. This allows LSTM to effectively use historical information so that it can learn long dependencies [35].

After input and calculation, the output layer can get a probability distribution of the flow classification $\{p_1, p_2, \ldots, p_m\}$. We define $p_{\max} = \max\{p_1, p_2, \ldots, p_m\}$ that determines the prediction category.

Finally, protocol types of encrypted traffic are obtained. We sent this m encrypted application traffic to the next module.

3.3. Second Label Module. On the basis of known encryption protocols, we leverage this module to further classify encrypted applications into n categories.

Corresponding to the m encryption protocols obtained in the last module, we prepare m classifiers. That is, each protocol corresponds to a classifier. encrypted traffic selects the corresponding classifier according to its protocol type, and each classifier is only responsible for the application classification of a specific protocol. By using different classifiers for different protocols, we can get more accurate results.

We choose CNN and LSTM in this module. In the end, we apply CNN. The performance of these two algorithms is addressed in Section 4.3.

4. Experiment and Evaluation

In this section, we do some experiments to evaluate ETCC and compare it with the state-of-the-art method. We deploy our model on Ubuntu 16.04 OS, equipped with NVIDIA GTX 1050 GPU.

4.1. Dataset Description. Three public datasets CICDarknet2020 [36], ISCXTor2016 [37], and ISCXVPN2016 [38]

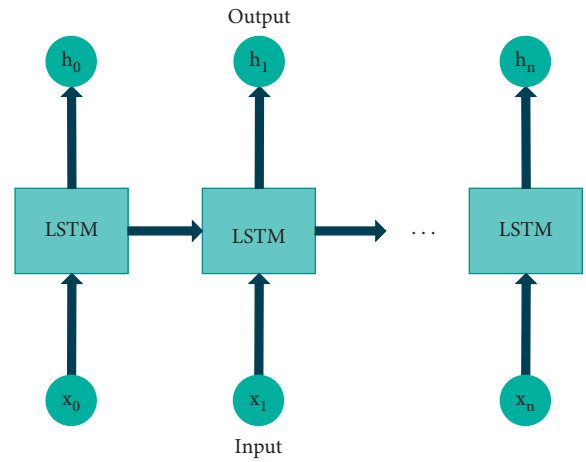


Figure 3: LSTM architecture.

are used to evaluate ETCC. These datasets include four types of protocols and five types of applications. The four protocols are Tor, Non-Tor, VPN, and Non-VPN. The five applications are chat, FTP, email, audio, and video, as shown in Table 2.

CICDarknet2020 is a complete dataset covering Tor traffic and VPN traffic. The specific quantity of each type of data is shown in Table 3. Since ISCXTor only has Tor traffic and ISCXVPN only has VPN traffic, we mix them together as a dataset, called ISCX-Tor-VPN. In order to eliminate errors caused by data sample selection, ISCX-Tor-VPN uses the same sample quantity as CICDarknet2020. In addition, we set the ratio of the train set to the test set with 4 : 1.

4.2. Parameter Settings. We deployed experiments for each classifier in each stage.

For the first label module, the structures of the CNN classifier and the LSTM classifier are shown in Figure 4. The dropout layer is used to discard neurons with a certain probability to prevent model overfitting and improve the generalization ability. Furthermore, we set the activation function, loss function, batch size, and epochs with ReLU, categorical_crossentropy, 32, and 15, respectively. For optimizer, the CNN classifier uses SGD, and the LSTM classifier uses Adam.

For the second label module, we have four classifiers to classify encrypted applications. The structures of the CNN classifier and the LSTM classifier are shown in Figure 5. Other parameters are the same as the last module.

Table 2: Encrypted applications details.

Traffic	Application
Chat	ICQ, AIM, Skype, Facebook, and Hangouts
FTP	Skype, SFTP, and FTPS
Email	SMTPS, POP3S, and IMAPS
Audio	Spotify
Video	YouTube and Vimeo

Table 3: The specific quantity of data.

Traffic	Chat	FTP	Email	Audio	Video
Tor	65	107	13	223	202
Non-Tor	410	6731	490	1469	3363
VPN	4476	2501	569	13060	1144
Non-VPN	6521	1795	5071	3296	4758

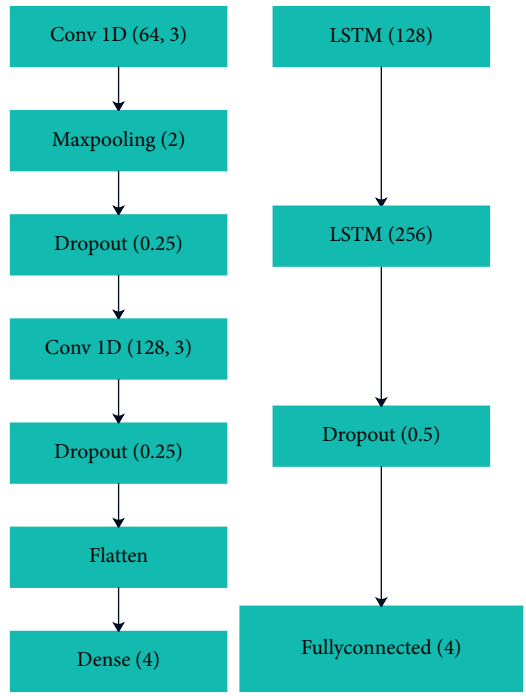


Figure 4: First label classifier's structure: (a) CNN structure and (b) LSTM structure.

4.3. Results and Discussion. In this section, we analyze the performance of ETCC on the two datasets and compare ETCC with the state-of-the-art method.

We evaluate the classification results after the first label module. Figure 6 shows confusion matrices of the results. Rows and columns represent the true category and predicted category. The value represents the probability of a category being classified into each category.

From Figure 6, we find that, under the same model, the results of CICDarknet2020 are better than the results of ISCX-Tor-VPN. This is because the data of CICDarknet2020 is generated under the same network environment, and ISCX-Tor-VPN is a mixed dataset, which makes the distinction between Tor and Non-Tor and between VPN and Non-VPN smaller. For the two classifiers, it is obvious that the results of CNN are better, so we choose CNN as the first stage classifier. In addition, we also find that the easily confused categories are VPN and Tor and Non-VPN and Non-Tor. It is not difficult to understand that there are some similar characteristics between encrypted traffic and non-encrypted traffic.

Tables 4 and 5 show the experimental results of the second label module on the premise that the first label module uses CNN classifier. Accuracy, precision, recall, and

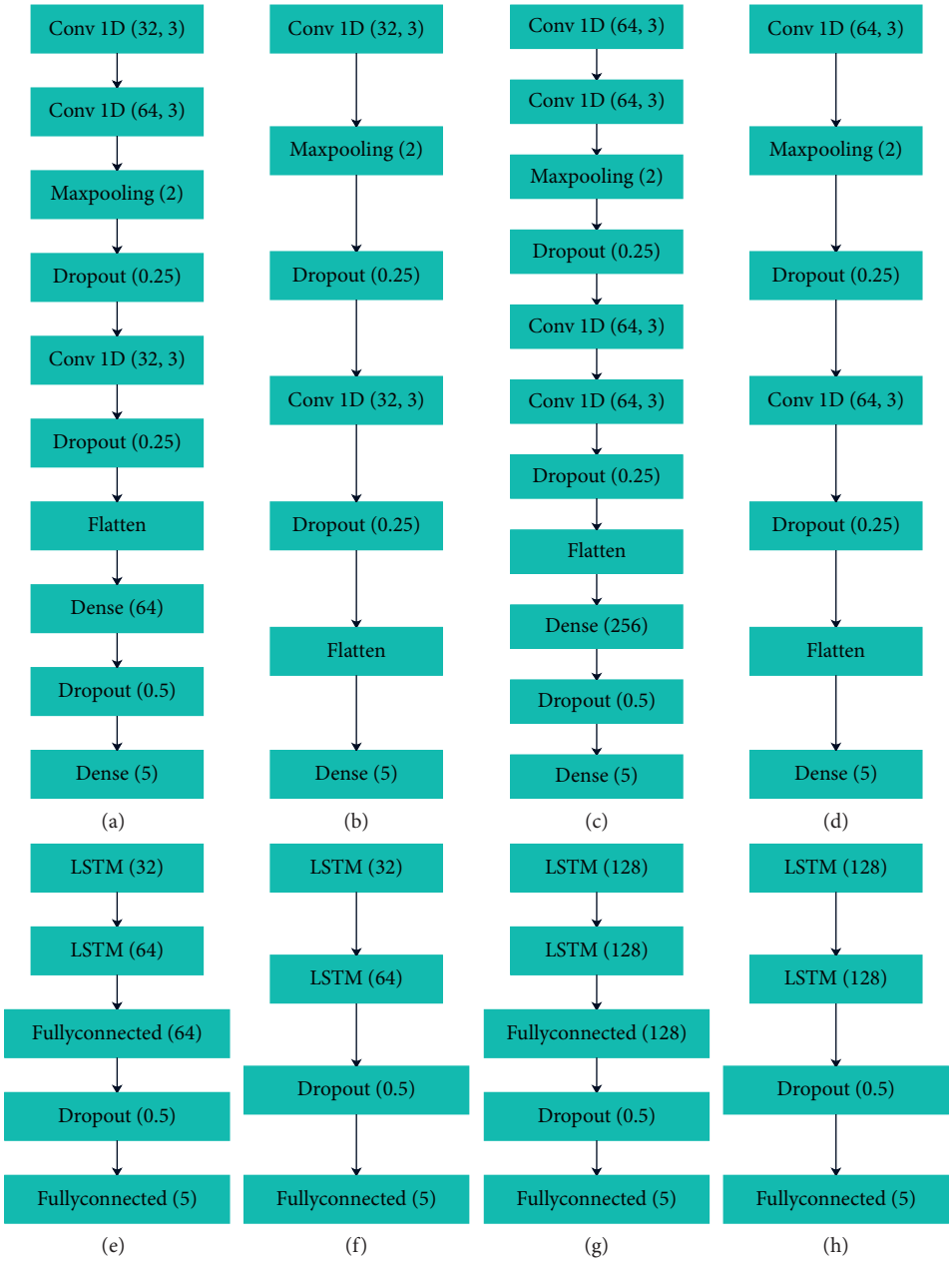


Figure 5: Second label classifier's structure: (a) CNN for Tor; (b) CNN for Non-Tor; (c) CNN for VPN; (d) CNN for Non-VPN; (e) LSTM for Tor; (f) LSTM for Non-Tor; (g) LSTM for VPN; (h) LSTM for Non-VPN.

F1 are used to evaluate the scheme. They are defined as follows:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{2}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{3}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{4}$$

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{5}$$

For category X, TP is the number correctly classified into X, TN is the number correctly classified into Not-X, FP is the number incorrectly classified into X, and FN is the number incorrectly classified into Not-X.

As can be seen from the Tables 4 and 5, CNN performs better than LSTM. For CICDarknet2020, except the F1 of Tor, other indicators CNN performs better. For ISCX-Tor-VPN, except the precision of Tor, the precision of Non-VPN, and the F1 of Non-VPN, other indicators CNN performs better. This is because CNN has a better understanding of local features, while LSTM can memorize some context information. In our dataset, the category of a flow has little relationship with the flow before and after it, so CNN

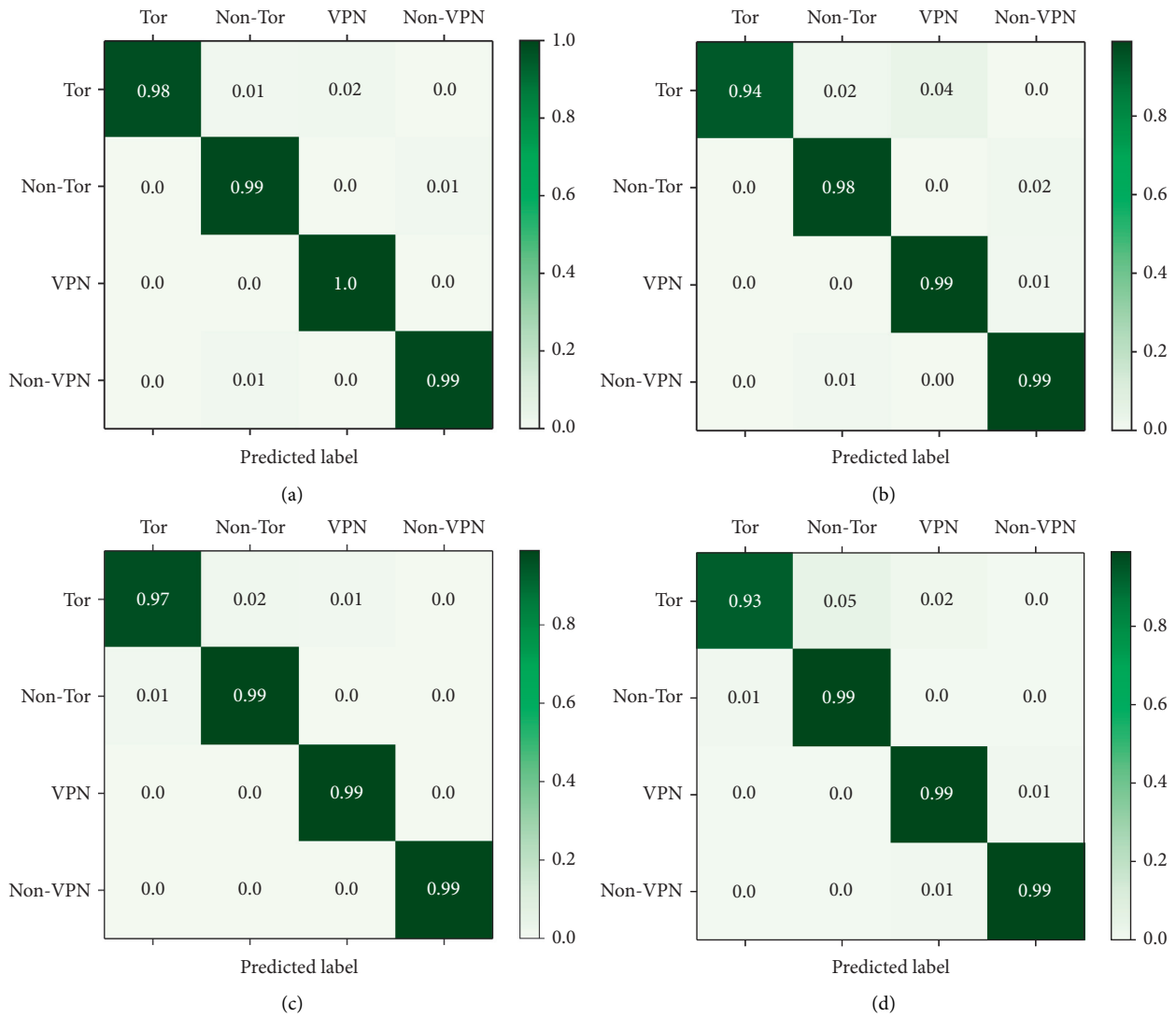


Figure 6: Confusion matrix with the first label module: (a) CNN classifier of CICDarknet2020; (b) LSTM classifier of CICDarknet2020; (c) CNN classifier of ISCX-Tor-VPN; (d) LSTM classifier of ISCX-Tor-VPN.

Table 4: The performance of CICDarknet2020 with the second label module (%).

Protocol	CNN				LSTM			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Tor	94.3	90.6	94.8	92.7	91.8	84.3	91.4	97.7
Non-Tor	98.8	97.0	98.3	97.6	98.6	96.3	97.8	97
VPN	99	97.3	98	97.6	98.8	96.9	97.6	97.2
Non-VPN	98.5	97.3	98.5	97.9	98.3	97.1	98.2	97.6

performs better. Therefore, we also chose CNN as the second stage classifier; the worst indicator also exceeds 91.1%.

Tables 6 and 7 show the performance with the second label module and CNN classifier. We find the classification capabilities of Non-Tor and Non-VPN classifiers are better than Tor and VPN classifiers. This proves that encryption makes traffic classification more difficult. Another observation is that the precision of email is very low; this is because the sample size of the email in the dataset is very small. This phenomenon will not occur when the sample size is balanced. Moreover, audio and video achieve the best classification results.

Finally, we compare the results of CICDarknet2020, ISCX-Tor-VPN and the state-of-the-art method [39], as shown in Table 8. The result of CICDarknet2020 is better than that of ISCX-Tor-VPN. The reason is as mentioned earlier; that is, ISCX-Tor-VPN is a mixed dataset, and data is less distinguishable. Moreover, compared with [39], except

Table 5: The performance of ISCX-Tor-VPN with the second label module (%).

Protocol	CNN				LSTM			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Tor	91.8	86.8	91.4	89.0	87.7	87.4	87.1	87.2
Non-Tor	97.7	94.3	96.9	95.6	97.2	93.1	96.1	94.6
VPN	98.3	96.3	97.1	96.7	98.1	95.6	96.7	96.1
Non-VPN	98.2	96.9	98.1	97.5	98	98.8	97.8	98.3

Table 6: The performance of CICDarknet2020 with the second label module and CNN classifier (%).

Application	Precision				Recall			
	Tor	Non-Tor	VPN	Non-VPN	Tor	Non-Tor	VPN	Non-VPN
Chat	92.3	94.1	99.7	99.8	92.3	97.6	98.3	98.4
FTP	95.2	99.5	95.3	89.6	90.9	99.3	98.4	98.4
Email	75	94.1	94.8	99.5	100	96.9	96.5	98.8
Audio	95.6	97.7	99.8	98.3	95.6	99	99.5	97.9
Video	95	99.5	97	99.5	95	98.5	97.4	98.8

Table 7: The performance of ISCX-Tor-VPN with the second label module and CNN classifier (%).

Application	Precision				Recall			
	Tor	Non-Tor	VPN	Non-VPN	Tor	Non-Tor	VPN	Non-VPN
Chat	84.6	86.8	99.5	99.7	84.6	96.3	97	98.3
FTP	86.4	99.1	91.4	88	86.4	98.6	98.2	98.1
Email	75	92.2	94.9	98.9	100	96.9	97.4	98.1
Audio	95.3	94.9	99.6	98.3	91.1	95.2	99.1	97.7
Video	92.7	98.5	96	99.5	95	97.3	93.9	98.3

Table 8: The comparison between CICDarknet2020, ISCX-Tor-VPN, and [39] (%).

Work	Chat		FTP		Email		Audio		Video		Total	
	P	R	*P*	R	*P*	R	*P*	R	*P*	R	*P*	R
CICDarknet2020	96.5	96.7	95	96.8	90.9	98	98	98	97.8	97.4	95.6	97.4
ISCX-Tor-VPN	92.7	94.1	91.2	95.3	90	98	97	96	96.7	96.1	93.6	95.9
Paper [39]	92	93	95	92	96	98	95	98	95	98	94.6	95.8

the precision of email and the recall of video, other indicators are improved. Total precision and recall increase by 1% and 1.6%, respectively. In general, our ETCC significantly improves the classification accuracy of encrypted applications through a two-stage two-label method. This proves that applications have different characteristics under different protocols, and the classification of applications on the basis of known protocols will result in more accurate results.

5. Conclusion and Future Work

In this paper, to achieve refined classification of encrypted applications, we propose a two-stage two-label scheme. The first stage classifies the protocol used for encrypted traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. The experimental results prove that our scheme is effective and feasible.

Furthermore, we discuss two-label classification in this paper. We will consider more labels in the future and propose more practical solutions. In addition, our method is based on the identification of encryption protocols. Once the traffic uses an unknown encryption protocol, the application classification results will be affected. Therefore, we will consider the use of unknown encryption protocols in our future work.

Data Availability

The datasets used in this paper are mainly obtained through the website https://www.unb.ca/cic/datasets/darknet2020.html; https://www.unb.ca/cic/datasets/tor.html; https://www.unb.ca/cic/datasets/vpn.html. The raw/processed data required to reproduce these findings cannot be shared at this time as the data also form part of an ongoing study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China under Grant no. 61702267, Jiangsu Planned Projects for Postdoctoral Research Funds, and in part supported by the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing.

References

[1] U. Cisco, Cisco Annual Internet Report (2018–2023) White Paper, 2020.

[2] A. Jakalan, J. Gong, Q. Su, X. Hu, and A. M. S. Abdelgder, "Social relationship discovery of IP addresses in the managed IP networks by observing traffic at network boundary," *Computer Networks*, vol. 100, pp. 12–27, 2016.

[3] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1257–1270, 2014.

[4] M. H. Mazhar and Z. Shafiq, "Real-time video quality of experience monitoring for https and quic,," in *Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 1331–1339, Honolulu, HI, USA, April 2018.

[5] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, "A survey of methods for encrypted traffic classification and analysis," *International Journal of Network Management*, vol. 25, no. 5, pp. 355–374, 2015.

[6] P. Wang, S. Li, F. Ye, Z. Wang, and M. Zhang, "PacketCGAN: exploratory study of class imbalance for encrypted traffic classification using CGAN," in *Proceedings of the ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–7, San Jose, USA, June 2020.

[7] 2020 Service name and transport protocol port number registry https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[8] F. Casino, K.-K. R. Choo, and C. Patsakis, "HEDGE: efficient traffic classification of encrypted and compressed packets," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 11, pp. 2916–2926, 2019.

[9] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2013.

[10] Y. Liu, J. Chen, P. Chang, and X. Yun, ""A novel algorithm for encrypted traffic classification based on sliding window of flow's first N packets," in *Proceedings of the 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*, pp. 463–470, Beijing, China, September 2017.

[11] H. F. Alan and J. Kaur, "Can Android applications be identified using only TCP/IP headers of their launch time traffic?" in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pp. 61–66, Darmstadt, Germany, July 2016.

[12] M. Shen, M. Wei, L. Zhu, and M. Wang, "Classification of encrypted traffic with second-order Markov chains and application attribute bigrams," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1830–1843, 2017.

[13] S. Cui, B. Jiang, Z. Cai, Z. Lu, S. Liu, and J. Liu, ""A session-packets-based encrypted traffic classification using capsule neural networks," in *Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications*, pp. 429–436, Zhangjiajie, China, August 2019.

[14] L. Vu et al., "Time Series Analysis for Encrypted Traffic Classification: A Deep Learning Approach," in *Proceedings of the 2018 18th International Symposium on Communications and Information Technologies (ISCIT)*, pp. 121–126, Austin, Texas, USA, September 2018.

[15] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "$Deep-Full-Range$: a deep learning based network encrypted traffic classification and intrusion detection framework," *IEEE Access*, vol. 7, pp. 45182–45190, 2019.

[16] E. Areström and N. Carlsson, "Early online classification of encrypted traffic streams using multi-fractal features," in *Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 84–89, Paris, France, May 2019.

[17] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pp. 43–48, Beijing, China, July 2017.

[18] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[19] M. Shen, J. Zhang, L. Zhu, K. Xu, X. Du, and Y. Liu, "Encrypted traffic classification of decentralized applications on ethereum using feature fusion," in *Proceedings of the 2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS)*, pp. 1–10, Passau, Germany, June 2019.

[20] I. Orsolic, D. Pevec, M. Suznjevic, and L. Skorin-Kapov, "A machine learning approach to classifying YouTube QoE based on encrypted network traffic," *Multimedia Tools and Applications*, vol. 76, no. 21, pp. 22267–22301, 2017.

[21] T. Mangla, E. Halepovic, M. Ammar, and E. Zegura, "Using session modeling to estimate HTTP-based video QoE metrics from encrypted network traffic," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 1086–1099, 2019.

[22] B. Anderson and D. McGrew, "Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1723–1732, New York, NY, August 2017.

[23] P.-O. Brissaud, J. Francois, I. Chrisment, T. Cholez, and O. Bettan, "Transparent and service-agnostic monitoring of encrypted web traffic," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 842–856, 2019.

[24] S. Rezaei and X. Liu, "How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets," 2018, https://arxiv.org/abs/1812.09761.

[25] G. Tsoumakas and I. Katakis, "Multi-label classification," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[26] T. T. Nguyen, T. T. T. Nguyen, A. V. Luong, Q. V. H. Nguyen, A. W.-C. Liew, and B. Stantic, "Multi-label classification via label correlation and first order feature dependance in a data stream," *Pattern Recognition*, vol. 90, pp. 35–51, 2019.

[27] J. Read, A. Bifet, G. Holmes, and B. Pfahringer, "Scalable and efficient multi-label classification for evolving data streams," *Machine Learning*, vol. 88, no. 1-2, pp. 243–272, 2012.

[28] D. K. Shah, M. A. Sanghvi, R. P. Mehta, P. S. Shah, and A. Singh, "Multilabel Toxic Comment Classification Using Supervised Machine Learning Algorithms," in *Proceedings of the Machine Learning for Predictive Analysis*, pp. 23–32, Springer, 2020.

[29] G. Ou, G. Yu, C. Domeniconi, X. Lu, and X. Zhang, "Multi-label zero-shot learning with graph convolutional networks," *Neural Networks*, vol. 132, pp. 333–341, 2020.

[30] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pp. 195–200, Bremen, Germany, October 2005.

[31] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 254–269, Bled, Slovenia, September 2009.

[32] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, and H. Wang, "SGM: Sequence Generation Model for Multi-Label Classification," 2018, https://arxiv.org/abs/1806.04822.

[33] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[34] S. Patro and K. K. Sahu, "Normalization: a preprocessing stage," 2015, https://arxiv.org/abs/1503.06462.

[35] Y. Li, Y. Lu, and L. S. T. M.-B. A. ", "DDoS Detection Approach Combining LSTM and Bayes," in *Proceedings of the 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD)*, pp. 180–185, Suzhou, China, September 2019.

[36] A. H. Lashkari, "DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic Using Deep Image Learning," in *Proceedings of the Presented at the International Conference on Communication and Network Security*, Tokyo, Japan, November 2020.

[37] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy*, pp. 253–262, Porto, Portugal, February 2017.

[38] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 407–414, Rome, Italy, February 2016.

[39] Z. Zou, J. Ge, H. Zheng, Y. Wu, C. Han, and Z. Yao, ""Encrypted traffic classification with a convolutional long short-term memory neural network," in *Proceedings of the 2018 IEEE 20th International Conference on High Performance Computing and Communications*, pp. 329–334, Exeter, UK, June 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6614702, 13 pages
https://doi.org/10.1155/2021/6614702

Research Article

Side-Channel Leakage Detection with One-Way Analysis of Variance

Wei Yang and Anni Jia

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Correspondence should be addressed to Wei Yang; generalyzy@njust.edu.cn

Received 1 January 2021; Revised 7 February 2021; Accepted 25 February 2021; Published 5 March 2021

Academic Editor: Liguo Zhang

Side-channel analysis (SCA) is usually used for security evaluation to test the side-channel vulnerability of a cryptographic device. However, in practice, an analyser may need to cope with enormous amounts of side-channel measurement data to extract valuable information for SCA. Under the circumstances, side-channel leakage detection can be used to identify leakage points which contain secret information and therefore improve the efficiency of security assessment. This investigation proposes a new black-box leakage detection approach on the basis of the one-way analysis of variance (ANOVA). In accordance with the relevance between leakage points and inputs of a cryptographic algorithm, the proposed method divides side-channel samples into multiple classes and tests the difference among these classes by taking advantage of the one-way ANOVA. Afterwards, leakage points and nonleakage points can be distinguished by determining whether the null hypothesis is accepted. Further, we extend our proposed method to multichannel leakage detection. In particular, a new SCA attack with a F-statistic-based distinguisher is capable of developing if the input of the leakage detection approach is replaced by a sensitive intermediate variable. Practical experiments show the effectiveness of the proposed methods.

1. Introduction

Side-channel analysis (SCA) applies the physical leakage signals (e.g., sound, power consumption, electromagnetic radiation, and the like) of a running cryptographic device for retrieving the secret information [1–4]. Due to the efficiency and easy achievement, SCA has been a serious threat to cryptographic devices, but at the same time, SCA can also be used to evaluate the side-cannel resistance of these devices. In practice, SCA usually needs numerous leakage traces for security evaluation. That may lead to a fairly high computation complexity and significantly reduce the efficiency of the assessment. Therefore, side-channel leakage detection is proposed to reduce the computation complexity of security evaluation and improve the efficiencies of SCA attacks [5, 6].

Points in a side-channel signal can be grouped into two sets: leakage points and nonleakage points [6]. Leakage points contain sensitive information related to the key used in a crypto device and can be utilized by SCA. By contrast, nonleakage points include useless information for SCA. The task of leakage detection is to find the leakage points in side-channel measurements. Generally, the existing leakage detection methods view leakage points as the outliers in the measurement data and use differing statistics to winnow them. A simple way is to exploit the SCA distinguisher output as the statistic; any point with a high score is assigned to the leakage point set [7]. Additionally, a major category of leakage detection is based on the hypothesis test and identifies the "outliers" through computing a test statistic and a significance level. For example, the typical Test Vector Leakage Assessment (TVLA) methodology searches the leakage points by exploiting *Welch*'s t-test. The TVLA checks the t-statistic over the same time instants. If the value of the statistic is greater than the threshold determined by a prespecified significance level, the corresponding point will be regarded as a leakage point. Some similar leakage detection approaches are also proposed, such as leakage detection based on mutual information [8], the correlation-based ρ-test [9], the paired t-test [10], leakage detection with χ^2-test [11], and the rest. Besides, a

method based on normalized interclass variance and a new approach based on the constant parameter channel model [12] are investigated as well.

Nevertheless, some of the aforementioned methods need numerous traces to highlight the difference between leakage points and nonleakage points [11, 13], some methods need the knowledge of leakage models [7, 13], some require the low-noise measurements [11] or carefully chosen inputs [5, 9, 10], some need complicated calculation [8, 12], and so forth.

In sight of the limitations of the previous work, this paper proposes a black-box leakage detection approach based on the one-way analysis of variance (ANOVA). The main idea of the proposed method is that side-channel leakage samples at one point depend on the corresponding inputs of a cryptographic algorithm. That is, the same input yields the same leakage, while two differing inputs lead to two differing leakages. Consequently, according to the relevance between the leakage points and inputs of the algorithm, side-channel samples can be divided into multiple groups. These groups corresponding to a leakage point are supposed to have a statically significant difference because that variations of the inputs and the corresponding side-channel leakages are synchronous. On the contrary, the groups corresponding to a nonleakage point should have a statistically insignificant difference. Then, leakage points can be detected by the one-way ANOVA which is used for testing the difference of multiple groups.

The proposed method relies only on the inputs (i.e., plaintext or ciphertext) of the cryptographic algorithm and the corresponding side-channel samples. Hence, it is unnecessary to acquire any in-depth knowledge of the cryptographic algorithm implementation and the leakage model of a device. Moreover, it has no special requirements for the inputs and measurements and can be performed rapidly.

Further, we develop a new SCA distinguisher by changing the input of the proposed leakage detection approach. The effectiveness of the SCA attack using the distinguisher is verified by two practical experiments.

The rest of the paper consists of four sections, including the preliminaries (Section 2), the proposed method and its extensions (Section 3), the experimental results and analysis (Section 4), and the conclusion (Section 5).

2. Preliminaries

2.1. Side-Channel Leakage Detection. Side-channel leakage detection is used to detect the information leakage in a specific side-channel [5]. Unlike SCA attacks, leakage detection needs not to distinguish the correct key used in a cryptographic implementation from the other incorrect key guesses. Its task is to find key-dependent sample points (i.e., leakage points) in a side-channel leakage trace. That is to say, the output of leakage detection is a set of points of interest, which are corresponding to the formative moments of leakages related to secret information. Leakage detection is an important part of the side-channel evaluation and provides a preliminary result for further analysis or advanced assessment.

2.2. Test Vector Leakage Assessment. Test Vector Leakage Assessment (TVLA) methodology is a classical and extensively used technology for side-channel leakage detection. The basic assumption of TVLA is that leakage points and nonleakage points belong to two normally distributed populations, respectively. Then, an analyser can distinguish leakage points from nonleakage points by using *Welch*'s t-test to compare the parameters of these two populations.

The TVLA methodology includes two methods, that is, specific t-test and nonspecific t-test [5, 13, 14]. For the specific t-test, the side-channel traces are divided into two sets in accordance with the value of a single bit of a targeted intermediate key-dependent variable. If the two trace sets are considered as two normally distributed populations, *Welch*'s t-test can be applied to detect side-channel information leakage by testing the difference of the two population means over each time instant in the traces. The t-statistic is shown as follows:

$$t = \frac{\left(\overline{T_A} - \overline{T_B}\right)}{\sqrt{\left(S_A^2/N_A\right) + \left(S_B^2/N_B\right)}}, \tag{1}$$

where T_A and T_B are the two trace sets, $\overline{T_A}$, $\overline{T_B}$, S_A, S_B, N_A, and S_B are the means, variances, and size of T_A and T_B, respectively. If the value of t-statistic at one time instant exceeds the threshold determined by the degrees of freedom and a prespecified significance level, the point will be categorized as a leakage point. Otherwise, the point will be categorized as a nonleakage point.

For the nonspecific t-test, it needs to acquire two groups of side-channel measurements with the same size to apply for the pairwise t-test. One set of traces corresponds to a fixed input, and the other corresponds to random inputs. If one point in time for which the value of t-statistic is beyond that threshold, the point will be selected as a leakage point. If the corresponding t-statistic is within the bounds determined by that threshold, the point will be marked as a nonleakage point.

2.3. Leakage Detection with χ^2-Test. The leakage detection with the χ^2-test can be regarded as a complement to TVLA [11]. The core idea of this approach is to apply the χ^2-test of independence to check the association between the input groups (i.e., plaintext or ciphertext) and the measured leakage points.

The χ^2-test of independence is also named the χ^2-test of association. It is a nonparametric test to determine whether two categorical variables are related or independent. The statistic for the χ^2-test of association is computed as

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{s}\left[\frac{\left(V_{ij} - NP_{ij}\right)^2}{NP_{ij}}\right], \tag{2}$$

where r represents the number of the input groups (i.e., plaintext or ciphertext), s represents the number of the measured leakage points, N denotes the total number of the observations of two categorical variables, V_{ij} and P_{ij} denote the number and the frequency of concurrence of the observations of two categorical variables, respectively.

Similarly, if one point in time for which the value of χ^2-test exceeds the threshold determined by the degrees of freedom and a prespecified significance level, the point will be incorporated into the leakage points. Otherwise, the point will be incorporated into the nonleakage points.

2.4. One-Way Analysis of Variance. One-way analysis of variance (one-way ANOVA) can be seen as a generalization of *Welch*'s t-test. Where *Welch*'s t-test would be applied for comparison of group means of two normally distributed populations, one-way ANOVA is applied for the comparison of multiple group means [15, 16]. It assumes that samples in each group are independently drawn from a normally distributed population, and all populations have the same variance. The test statistic is

$$F = \frac{(N-1)SS_b}{[(r-1)SS_w]}, \tag{3}$$

where r represents the number of the input groups, N denotes the total number of the samples in all groups, SS_b denotes the sum of squares between groups, that is, the *between-group variation*, and SS_w denotes the residual variance within groups, that is, the *within-group variation*. If the value of F-statistic is beyond the threshold, it states that all r populations own the same mean value.

3. The Proposed Method

In this section, one-way ANOVA is applied for leakage detection, including monochannel and multichannel leakage detection. In addition, a novel SCA distinguisher is developed to serve for key recovery attack as well.

3.1. Side-Channel Leakage Detection with One-Way ANOVA. What the one-way ANOVA investigates is the influence of a numerical or categorical input variable on a numerical dependent variable. For a cryptographic implementation, the side-channel leakage samples depend on the inputs and the secret key of the crypto algorithm (i.e., plaintext or ciphetext). Since the key is usually changeless in a short period of time, the leakage data can be viewed as the numerical response data of a variable which relies on the inputs. Additionally, the possible values of the plaintext or ciphertext of a crypto algorithm are limited and can be categorized into multiple classes. Therefore, it is possible to detect leakage points by exploiting the one-way ANOVA.

Suppose that there are N side-channel leakage traces. They are collected from a running crypto device and correspond to N random plaintext (or ciphertext) with the same key. At one point in time, these N traces can be grouped into r groups according to the inputs of the crypto algorithm, and the i^{th} group consists of n_i samples, scilicet $l_{i1}, l_{i2}, \ldots, l_{in_1}$, $i = 1, 2, \ldots, r$. Let $l_1, l_2, \ldots, l_r$ denote the normally distributed populations corresponding to the r groups, respectively. That is,

$$l_{ij} \sim \phi\left(\mu_i, \sigma^2\right), \tag{4}$$

where $\phi(\mu_i, \sigma^2)$ represents the normal distribution with mean μ_i and variance σ^2, $i = 1, 2, \ldots, r$, $j = 1, 2, \ldots, n_i$.

For a leakage point in time, the corresponding leakage should be varied with the input variable of the crypto algorithm. Consequently, at this point, the leakage samples in all r groups drawn from the r populations should be statistically distinguishable. In other words, the means of these populations should have a significant difference. On the contrary, for a nonleakage point, the corresponding samples contain no information about the secret key. As a result, the variation of the samples should be random at this point. In this case, the samples in all r groups drawn from the r populations should be statistically indistinguishable. Namely, the means of these populations should have a statistically insignificant difference.

On the basis of the above-mentioned inference, we can perform leakage detection by using the one-way ANOVA. Table 1 depicts the data organization of the one-way ANOVA. Hence, the between-group variation SS_b and the within-group variation SS_w in equation (3) can be, respectively, rewritten as

$$SS_b = \sum_{i=1}^{r} \left(\overline{l_i} - \sum_{i=1}^{r} \frac{n_i \overline{l_i}}{n} \right)^2, \tag{5}$$

$$SS_w = \sum_{i=1}^{r} \sum_{j=1}^{n_i} \left(l_{ij} - \overline{l_i} \right)^2, \tag{6}$$

where $\overline{l_i}$ means the sample mean of the i^{th} group. Obviously, the expectation of $\overline{l_i}$ is equal to μ_i. It can be viewed that the between-group variation SS_b indicates the difference between each group mean and the total sample mean, and the within-group variation SS_w indicates the difference between the samples in a group and the group mean. If SS_b is much greater than SS_w, it states that there is a significant difference among all groups, and these r populations are unlikely from the same normal population. From this viewpoint, it is reasonable to apply the one-way ANOVA for leakage detection, because the samples corresponding to leakage points contain information about multiple input data and can be deemed to come from different populations, while the samples corresponding to nonleakage points include random data information and can be considered to come from the same population.

The threshold for the hypothesis test is

$$\text{Th} = F_{1-\alpha}(r-1, N-1), \tag{7}$$

where α denotes the significance level and F and $(r-1, N-1)$ represent the F distribution and the degrees of freedom, respectively. Points which exceed the threshold Th are acceptable as the leakage points, while the other points within the bound

Table 1: One-way ANOVA data organization.

Group	Observations of samples	Sample mean	Sample variance
1	$l_{11}, l_{12}, \ldots, l_{1n_1}$	$\overline{l_1}$	S_1^2
2	$l_{21}, l_{22}, \ldots, l_{2n_2}$	$\overline{l_2}$	S_2^2
3	$l_{31}, l_{32}, \ldots, l_{3n_3}$	$\overline{l_3}$	S_3^2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$r-2$	$l_{r-2,1}, l_{r-2,2}, \ldots, l_{r-2,n_{r-2}}$	$\overline{l_{r-2}}$	S_{r-2}^2
$r-1$	$l_{r-1,1}, l_{r-1,2}, \ldots, l_{r-1,n_{r-1}}$	$\overline{l_{r-1}}$	S_{r-1}^2
r	$l_{r1}, l_{r2}, \ldots, l_{rn_r}$	$\overline{l_r}$	S_r^2

determined by the threshold are classified as the nonleakage points. The whole test procedure is described as follows:

Initially, for each point in time, divide the side-channel samples at this point into multiple groups according to the corresponding plaintext or ciphertext.

Secondly, compute the value of F-statistic in accordance with equations (3), (5), and (6).

Finally, compare the F-statistic with the threshold Th in equation (7), and mark the point as a leakage point ($F > \mathrm{Th}$) or a nonleakage point ($F \leq \mathrm{Th}$).

The proposed leakage detection algorithm (LD_ANOVA for short) is elaborated in Algorithm 1. The symbol L denotes a matrix constructed by side-channel traces in row, X denotes the corresponding plain text or cipher text set, and α denotes the significance level. Further, L can be represented by the following formula:

$$L = \bigcup_{m=1}^{M} \bigcup_{i=1}^{r_m} \bigcup_{j=1}^{n_i} l_{ij}. \tag{8}$$

3.2. Extension to Multichannel Leakages. Generally, a crypto device in operation leaks multiple types of side information simultaneously. Consider that multichannel leakages usually contain much more potential key-dependent information than monochannel leakage [17], and it is feasible for an analyser to collect these leakages from multiple channels in practice; we also investigate how to enhance the leakage detection result by utilizing multichannel leakages.

To make the proposed detection approach in the previous section apply to the case of multichannel leakages as well, a preprocessing step aiming at transforming the multichannel leakage sets into a fused leakage set is needed. Based on the types of crypto implementations, there are different methods to perform the transform [17]. For instance, it is suitable to use the treatment of *Simple Fusion Attack* (SFA) to perform transform, if multichannel measurements are acquired from a crypto implementation with a few leakage points. And the fusion way of Fusion Attack Based Singular Value Decomposition (SVD_FA) may be a better fit for an implementation with multiple leakage points [17].

Algorithm 2 describes the proposed leakage detection algorithm for multichannel leakage detection. In the algorithm, the pseudocode in the first row means that, according to the leakage characteristic of a crypto implementation, applying a function Tr to transform the multichannel leakage sets ML into a monochannel leakage set L and transform the multichannel input sets MX into a monochannel input set X corresponding to L. The operator $\mathrm{Tr}(\cdot)$ indicates the corresponding transform.

3.3. SCA Distinguisher Based on One-Way ANOVA. Interestingly, we find that if we group the side-channel leakage samples in Table 1 in accordance with the values of a targeted intermediate variable, we can develop a novel SCA distinguisher to perform key recovery.

The idea is that the difference of means among all groups is supposed to be the most significant when the leakage data is categorized correctly. As compared to the correct key, the wrong key candidates will make the samples in the same group scattered in disparate groups and therefore make the difference of all populations indistinguishable and the leakage information related to the correct key hidden. Hence, the SCA distinguisher can be defined as follows:

$$\Delta \xrightarrow{\text{yields}} \widehat{k} \rightleftharpoons F(\widehat{k}) = \max_{k} F(k). \tag{9}$$

In equation (9), k and $\widehat{k}$ denote the candidates of key and the key guess, respectively. In addition, $F(\cdot)$ represents the F-statistic function in equation (3).

Naturally, a new SCA method yields. Its details are shown in Algorithm 3.

4. Experimental Results and Discussion

4.1. Monochannel Leakage Detection. To access the effectiveness of the proposed detection approach (i.e., LD_ANOVA), we collected two sets of side-channel traces to perform monochannel leakage detection at first. These two side-channel leakage sets include 30, 000 power traces acquired from an unprotected AES-128 encryption algorithm running on FPGA (Xilinx Kintex-7) and 5, 000 power traces acquired from a masked AES-128 encryption algorithm running on an embedded 8-bit MCU. The FPGA implementation belongs to a parallel hardware implementation, while the MCU version is a serial software implementation. The traces are all corresponding to random inputs. Parameters of the leakage acquisition are listed in Table 2.

Figures 1 and 2 show the detection results of the FPGA implementation by the proposed method and the other two typical detection methods. The two selected approaches used for comparison are the TVLA technology [13] and its complement, namely, the leakage detection with the χ^2-test (LD_CS for short) [11]. Compared with the nonspecific t-test, the specific t-test can offer higher confidence of detection at a lower cost [18]; the paper uses the latter t-test as the comparison. All values of the significance level α in the three aforementioned methods for leakage detection are set to 0.0001 to obtain results at a confidence of 99.99% [5]. Note that, the red-dashed lines in all figures represent the thresholds determined by the significance level α.

It can be seen that our proposed approach performs better than the other methods. For instance, LD_ANOVA is capable of finding the first leakage point by exploiting 10, 000 traces. In this case, both the TVLA and LD_CS fail to

Input: L, X, and α
Output: leakage point set LP
(1) $LP \longleftarrow \varnothing$
(2) **For** $m = 1, 2, \ldots, M$ // M denotes the number of points per trace
(3) $SS_b \longleftarrow \varnothing$, $SS_w \longleftarrow \varnothing$, $F \longleftarrow \varnothing$, $\mathrm{Th} \longleftarrow \varnothing$, $\overrightarrow{\mu} \longleftarrow \varnothing$
(4) Divide the m^{th} column of L into r_m groups according to the samples of X
(5) **For** $i = 1, 2, \ldots, r_m$
(6) $\overrightarrow{\mu} \longleftarrow \overrightarrow{\mu} \cup (\sum_{j=1}^{n_i} l_{ij}/n_i)$
(7) **End**
(8) $SS_b \longleftarrow \sum_{i=1}^{r_m} (\overrightarrow{\mu_i} - \sum_{i=1}^{r_m} n_i \overrightarrow{\mu_i}/n)^2$ $SS_w \longleftarrow \sum_{i=1}^{r_m} \sum_{j=1}^{n_i} (l_{ij} - \overline{l_i})^2$ // $\overrightarrow{\mu_i}$ denotes the j^{th} element of the vector $\overrightarrow{\mu}$.
(9) $F \longleftarrow (N-1)SS_b/[(r-1)SS_w]$
(10) $\mathrm{Th} \longleftarrow F_{1-\alpha}(r-1, N-1)$
(11) $LP \longleftarrow LP \cup t_m$, iff $F > \mathrm{Th}$
(12) **End**
(13) **Return**: LP.

Algorithm 1: Leakage detection based on one-way ANOVA.

Input: ML, MX, and α
Output: leakage point set LP
(1) $L \leftarrow \mathrm{Tr}(ML)$, $X \leftarrow \mathrm{Tr}(MX)$
(2) On the basis of X and α, apply Algorithm 1 to detect leakage points
Return: LP.

Algorithm 2: Leakage detection with multichannels leakages.

Input: L, X, and K
Output: key guess $\widehat{k}$
(1) Select a targeted intermediate variable
(2) **For** $i = 1, 2, \ldots, |K|$
(3) Calculate the intermediate values in accordance with the samples of X
(4) $F_0 \longleftarrow \varnothing$
(5) **For** $m = 1, 2, \ldots, M$
(6) $SS_b \longleftarrow \varnothing$, $SS_w \longleftarrow \varnothing$, $\overrightarrow{\mu} \longleftarrow \varnothing$
(7) Divide the m^{th} column of L into r_m groups according to the target intermediate values
(8) **For** $i = 1, 2, \ldots, r_m$
(9) $\overrightarrow{\mu} \longleftarrow \overrightarrow{\mu} \cup (\sum_{j=1}^{n_i} l_{ij}/n_i)$
(10) **End**
(11) $SS_b \longleftarrow \sum_{i=1}^{r_m} (\overrightarrow{\mu_i} - \sum_{i=1}^{r_m} n_i \overrightarrow{\mu_i}/n)^2$ $SS_w \longleftarrow \sum_{i=1}^{r_m} \sum_{j=1}^{n_i} (l_{ij} - \overline{l_i})^2$
(12) $F_0 \longleftarrow F_0 \cup (N-1)SS_b/[(r-1)SS_w]$
(13) **End**
(14) $F(k) \longleftarrow \max\{F_0\}$
(15) **End**
(16) $\widehat{k} \longleftarrow \max_k F(k)$
Return: $\widehat{k}$

Algorithm 3: SCA distinguisher based on one-way ANOVA.

detect any leakage point. Moreover, LD_ANOVA always identifies more leakage points than the TVLA and LD_CS. A comparison for the number of leakage points detected by these three methods is listed in Table 3 (rows 2, 6, and 10). In order to accurately evaluate the efficiency of the proposed method, the false positive points (i.e., the detected non-leakage points, denoted as N_{fp}) and the false negative points (i.e., the undetected leakage points, denoted as N_{fn}) are shown in Figure 3. Likewise, the number of the detected "real" leakage points are investigated (denoted as N_c).

Table 2: Parameters of the leakage acquisition.

Crypto implementation	Sample rate	Number of samples	Number of traces	Leakage model
FPGA	5G Sa/s	2,500	30,000	Hamming Distance
MCU	1G Sa/s	20,000	5,000	Hamming Weight

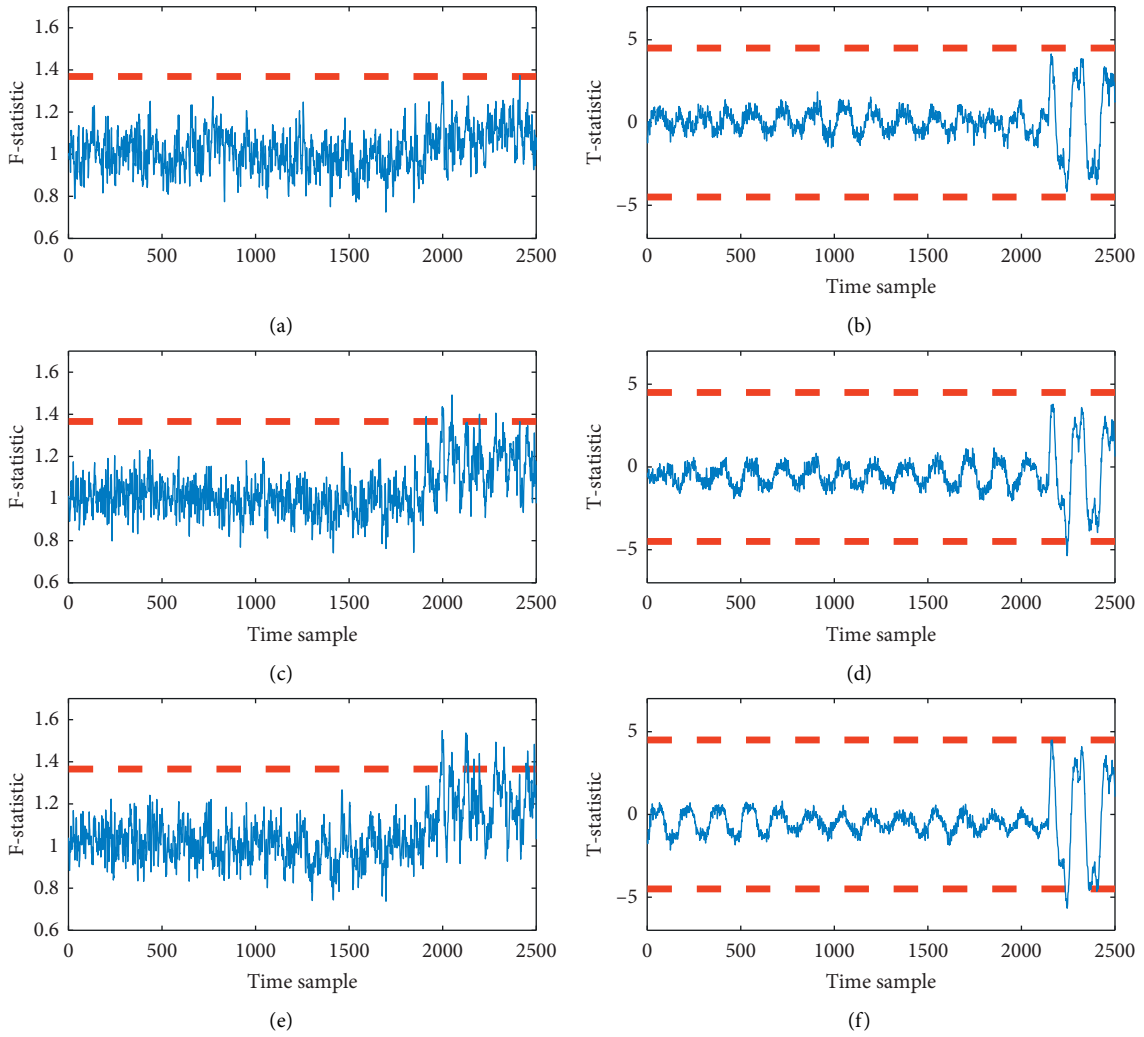


Figure 1: Detection results of TVLA and LD_ANOVA for the FPGA implementation. Left column: results of LD_ANOVA. Right column: results of TVLA. Upper row: 10,000 used traces. Middle row: 20,000 used traces. Lower row: 30,000 used traces.

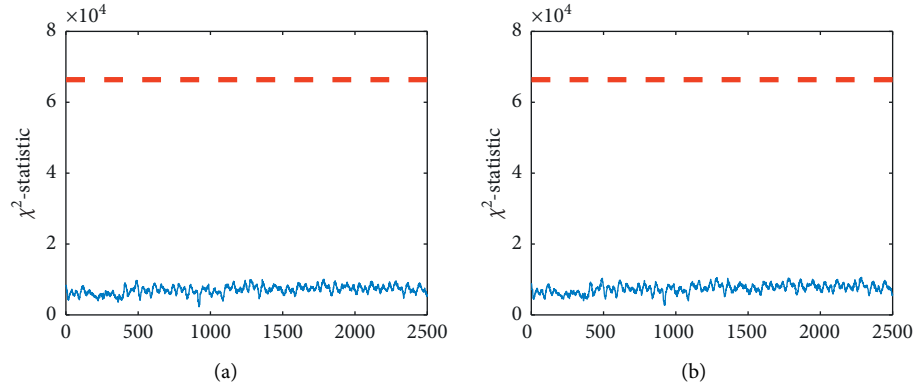
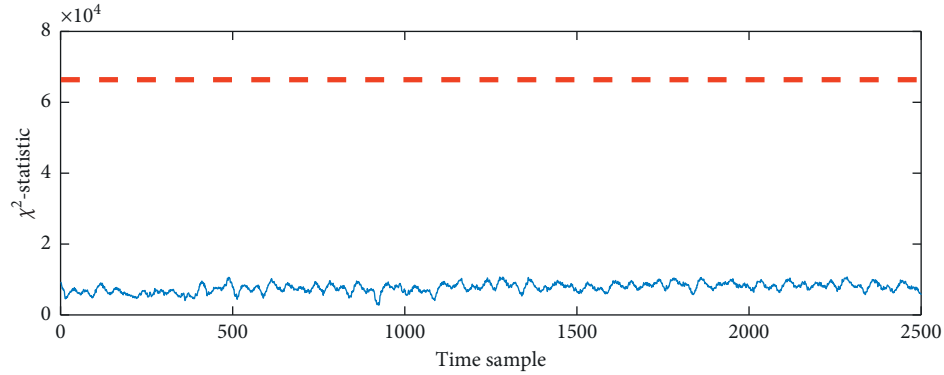


Figure 2: Continued.



(c)

Figure 2: Detection results of LD_CS for the FPGA implementation. Upper row (left column): 10,000 used traces. Upper row (right column): 20,000 used traces. Lower row: 30,000 used traces.

Table 3: Detection results of the FPGA implementation.

Detection method		10,000 traces	20,000 traces	30,000 traces
LD_ANOVA	D	1 point	21 points	68 points
	N_{fp}/R_{fp}	0/0%	4/2.72%	7/4.76%
	N_{fn}/R_{fn}	146/99.32%	130/88.44%	86/58.50%
	N_c/R_c	1/0.68%	17/11.56%	61/41.50%
TVLA	D	0 points	7 points	21 points
	N_{fp}/R_{fp}	0/0%	3/2.04%	3/2.04%
	N_{fn}/R_{fn}	147/100%	143/97.28%	129/87.76%
	N_c/R_c	0/0.0%	4/2.72%	18/12.24%
LD_CS	D	0 points	0 points	0 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	147/100%	147/100%	147/100%
	N_c/R_c	0/0%	0/0%	0/0%

Naturally, the coverage rate of the "real" leakage points, the false positive rate, and the false negative rate are also calculated and illustrated in Figure 3. These three metrics are defined as follows:

$$R_{fp} = \frac{|D - D \cap P|}{|P|} \times 100\%, \tag{10}$$

$$R_{fn} = \frac{|P - D \cap P|}{|P|} \times 100\%, \tag{11}$$

$$R_c = 100\% - R_{fn} = \frac{|D \cap P|}{|P|} \times 100\%, \tag{12}$$

where P means the leakage point set of an implementation; D means the detected leakage point set; "$|\cdot|$" denotes the cardinality of a set; and R_{fp}, R_{fn}, and R_c represent the false positive rate, the false negative rate, and the coverage rate, respectively. It is noted that an efficient detection method should lead to lower R_{fp} and R_{fn} and higher R_c as compared to an inefficient method. From the data in Table 3, it can draw a conclusion that the proposed method is more efficient than the other two.

For the masked implementation, before applying for the detection approaches, the side-channel samples are recombined by the normalized product function to eliminate the influence of Boolean masking [19]. The preprocessing function computes the product of the leakages corresponding to the masked intermediate variable and the masking variable as the leakage of the key-dependent variable so that leakage points can be matched with the inputs. Figures 3 and 4 illustrate that LD_ANOVA still exhibits the best performance. For example, it marks 38 points as leakage points when the TVLA finds one leakage point by using 1,000 traces. Although these points contain plenty of nonleakage points, there are still 20 leakage points. Due to the limited number of traces, the detection result is changeable; for example, only 4 leakage points are discovered when we use 3,000 traces to execute LD_ANOVA. The number of the detected leakage points increases with the increasing traces; for example, 39 leakage points are identified when 5,000 traces are used. The information of leakage points detected by the three methods is described in Table 4. The three rates defined in equations (10)–(12) are demonstrated in Table 4 as well. The results lead to a similar inference to the first experiment.

Besides, the coverage rate in Table 4 becomes higher than that in Table 3 when all the traces are used. The cause of this phenomenon is that the FPGA implementation is a parallel implementation which makes the collected traces include more switching noise [20] as compared to the 8-bit MCU implementation.

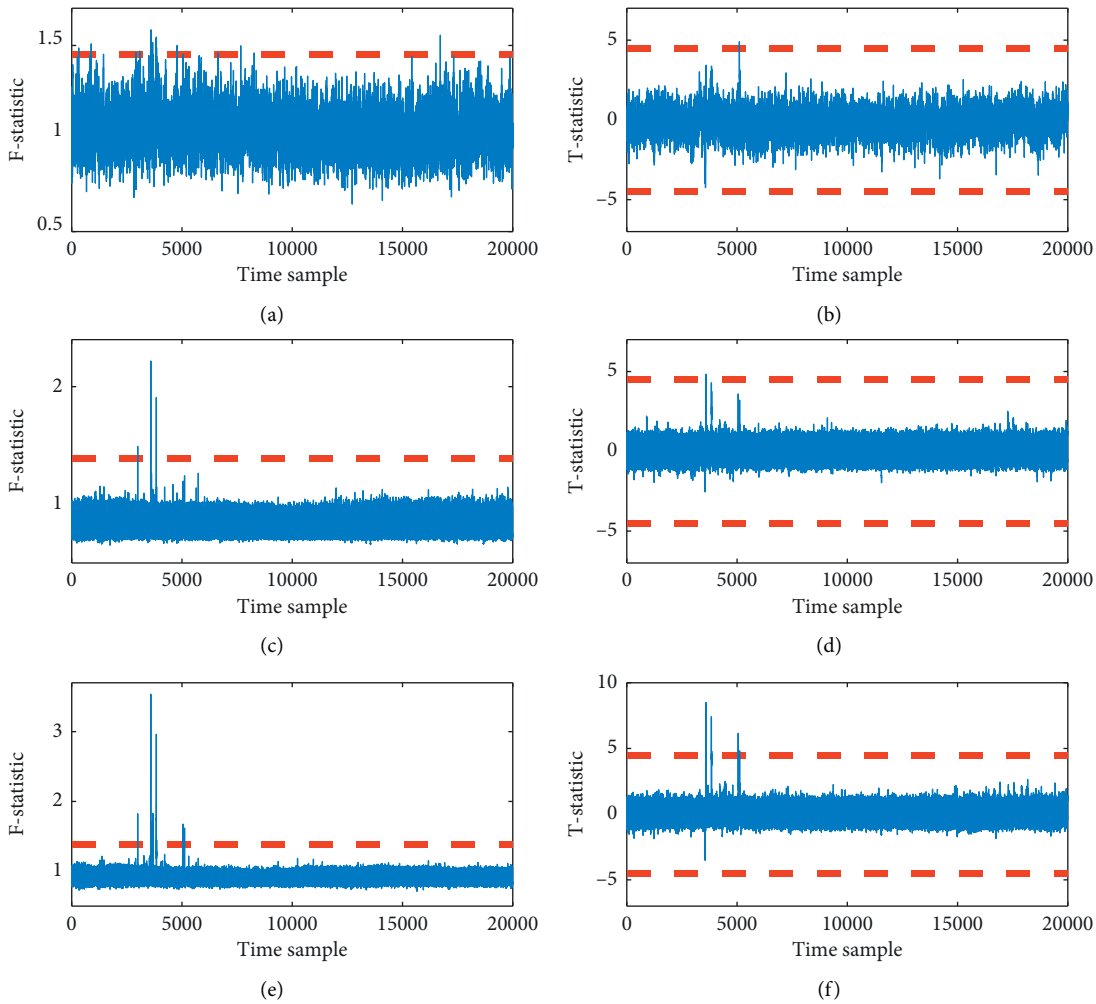


Figure 3: Detection results of TVLA and LD_ANOVA for the 8-bit MCU implementation. Left column: results of LD_ANOVA. Right column: results of TVLA. Upper row: 1,000 used traces. Middle row: 3,000 used traces. Lower row: 5,000 used traces.

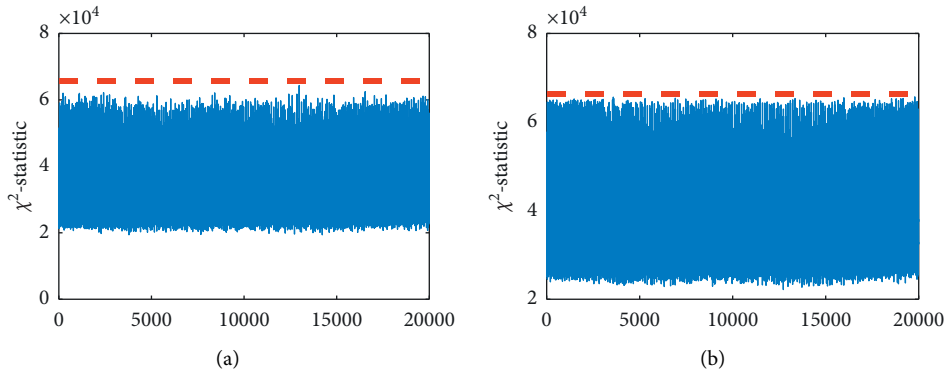
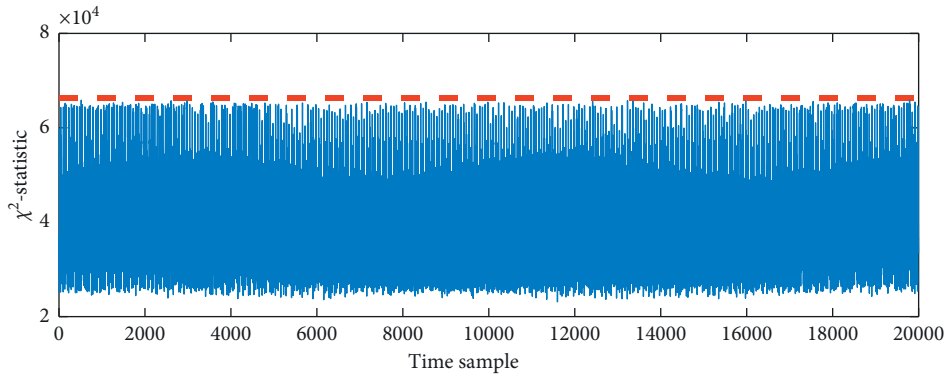


Figure 4: Continued.



(c)

Figure 4: Detection results of LD_CS for the 8-bit MCU implementation. Upper row (left column): 1,000 used traces. Upper row (right column): 3,000 used traces. Lower row: 5,000 used traces.

Table 4: Detection results of the 8-bit MCU implementation.

Detection method		1,000 traces	3,000 traces	5,000 traces
LD_ANOVA	D	38 points	4 points	39 points
	N_{fp}/R_{fp}	18/36.73%	0/0%	0/0%
	N_{fn}/R_{fn}	29/59.18	45/91.84%	10/20.41%
	N_c/R_c	20/41.82%	4/8.16%	39/79.59%
TVLA	D	1 point	1 point	28 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	48/97.96%	48/97.96%	21/42.86%
	N_c/R_c	1/2.04%	1/2.04%	28/57.14%
LD_CS	D	0 points	0 points	0 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	49/100%	49/100%	49/100%
	N_c/R_c	0/0%	0/0%	0/0%

It is noted that LD_CS fails to distinguish leakage points from nonleakage points in the above two experiments. The reason is that the method needs to divided leakage samples into multiple bins before performing the test of independence, and the process is vulnerable to the noise in the side-channel traces, which will influence the detection efficiency [12]. The reason that LD_ANOVA shows a better performance than the TVLA is because the former divides side-channel samples into more than two classes according to the input classes and obtains much more information about the relevance between the two.

4.2. Multichannel Leakage Detection. Consider that multiple types of leakages of a running cryptographic device can be collected from multiple side channels, which may expose much more information that monochannel SCA cannot utilize; the proposed method is used for multichannel leakage detection as well. We simultaneously collected 30,000 and 5,000 electromagnetic (EM) traces from the FPGA and MCU implementations in the previous section (Section A, Section 4), respectively. Due to the fact that the oscilloscope we used in the experiments can only collect the power traces and the EM traces with the same sample rate, the sample rates of these two EM trace sets are set to be the same as the sample rates of the corresponding power traces (viz., 5G Sa/s and 1G Sa/s). The sample rates are experimentally selected to ensure both the power and EM trace sets contain adequate leakage information for side-channel analysis.

Figure 5 depicts an example of multichannel leakage detection for the FPGA implementation. The fused trace set is obtained by the nonnegative matrix factorization-based leakage fusion [17]. Two monochannel detection results by utilizing the power leakage and the EM leakage are also illustrated in the figure. As shown in Figure 5, multichannel leakage detection performs better than any monochannel leakage detection.

Figure 6 compares the union of the detected power leakage points and the detected EM leakage points to the result of the multichannel leakage detection by exploiting fused traces. It can be viewed that the fusion of monochannel leakages makes a mass of potential monochannel leakage points exposed. For instance, 341 leakage points are identified when the multichannel detection uses all fused traces, while the monochannel detection only identifies 50 and 61 leakage points, respectively. Furthermore, the fusion of the differing channel makes the influence of the switching noise

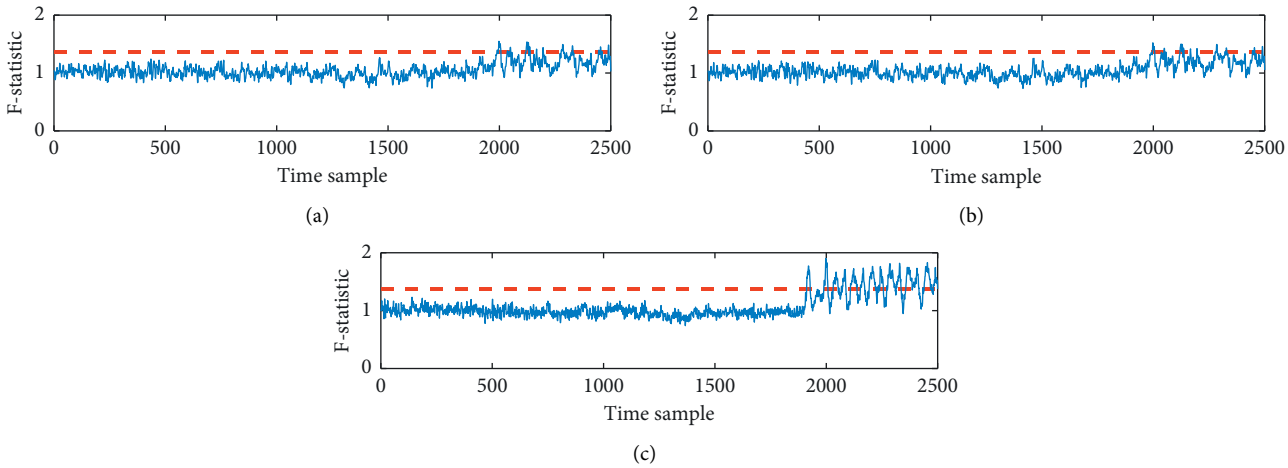


Figure 5: Results of monochannel leakage detection and multichannel leakage detection by LD_ANOVA for the FPGA implementation. Upper row: detected power leakage points (10,000 used traces). Middle row: detected EM leakage points (20,000 used traces). Lower row: detected fused leakage points (30,000 used traces).

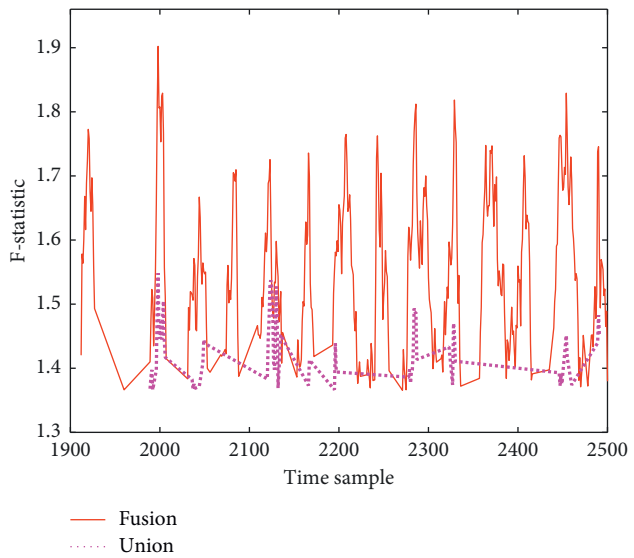


Figure 6: Detected leakage points of fused traces and union of detected monochannel leakage points (the FPGA implementation). "Union": union of detected power and EM leakage point sets. "Fusion": detected leakage points of fused traces.

in the traces significantly reduce and the coverage rate significantly increase. More detailed data is illustrated in Table 5.

The detection results for the 8-bit MCU implementation draw a similar conclusion. Similarly, the corresponding details are described in Table 6.

4.3. Runtime and Memory. To ensure that our proposed method can be efficiently performed for practical evaluation, the runtime and memory of the proposed approach are investigated experientially. We perform a monochannel leakage detection by applying TVLA, LD_CS, and LD_ANOVA, respectively.

This experiment is executed on the HP laptop with the Intel Core i7-7500U CPU (Dual-Core, 2.70 GHz and 2.90 GHz), 8 GB RAM, and MATLAB R2017a software platform. The three detections are repeated 10 times with 30,000 power traces acquired from the FPGA implementation in Section A, and the averages of the runtimes and the memories are plotted in Figure 7.

Note that, the code of the TVLA approach is performed in parallel. As a result, it can be seen that TVLA is the most efficient in terms of time. Moreover, LD_ANOVA is the most efficient in terms of memory, while LD_CS is the most inefficient—in both time and memory. On the whole, the computation cost of a practical security evaluation applying the proposed leakage detection approach is able to afford an analyser.

4.4. New Attack. Finally, two practical experiments are also carried out to evaluate the effectiveness of the proposed new

Table 5: Monochannel and multichannel leakage detection results for the FPGA implementation.

Leakage		10,000 traces	20,000 traces	30,000 traces
Power traces	D	1 point	21 points	68 points
	N_{fp}/R_{fp}	0/0%	4/2.72%	7/4.76%
	N_{fn}/R_{fn}	146/99.32%	130/88.44%	86/58.50%
	N_c/R_c	1/0.68%	17/11.56%	61/41.50%
EM traces	D	0 points	19 points	52 points
	N_{fp}/R_{fp}	0/0%	0/0%	2/1.64%
	N_{fn}/R_{fn}	122/100%	103/84.43%	72/59.02%
	N_c/R_c	0/0%	19/15.57%	50/40.98%
Fused traces	D	4 points	168 points	355 points
	N_{fp}/R_{fp}	0/0%	8/2.35%	14/4.11%
	N_{fn}/R_{fn}	337/98.83%	181/53.08%	0/0%
	N_c/R_c	4/1.17%	160/46.92%	341/100%

Table 6: Monochannel and multichannel leakage detection results for the 8-bit MCU implementation.

Leakage		1,000 traces	3,000 traces	5,000 traces
Power traces	D	38 points	4 points	39 points
	N_{fp}/R_{fp}	18/36.73%	0/0%	0/0%
	N_{fn}/R_{fn}	29/59.18	45/91.84%	10/20.41%
	N_c/R_c	20/41.82%	4/8.16%	39/79.59%
EM traces	D	24 points	3 points	24 points
	N_{fp}/R_{fp}	16/43.24%	0/0%	0
	N_{fn}/R_{fn}	29/78.38%	34/91.89%	13/35.14%
	N_c/R_c	8/21.62%	3/8.11%	24/64.86%
Fused traces	D	15 points	54 points	62 points
	N_{fp}/R_{fp}	2/3.13%	2/3.13%	0/0%
	N_{fn}/R_{fn}	51/79.69%	12/18.75%	2/3.13%
	N_c/R_c	13/20.31%	52/81.25%	62/96.87%

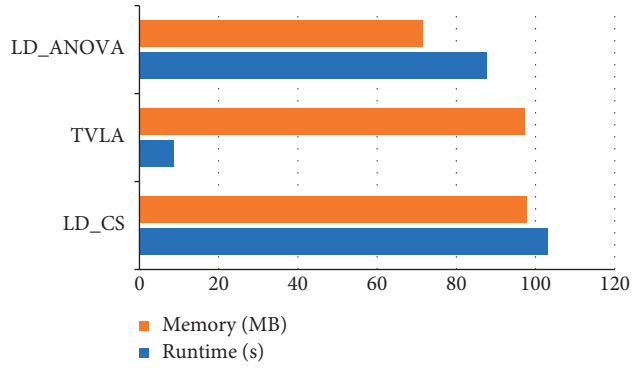


Figure 7: Runtimes and memories of LD_ANOVA, TVLA, and LD_CS.

SCA attack based on ANOVA (ANOVA_SCA for short) in Algorithm 3. Two power trace sets and two EM trace sets of the two aforementioned implementations in Section A are exploited to launch the key attack recovery, respectively. The leakage models of the FPGA and MCU implementations are approximate to the Hamming Distance and Hamming Weight models, respectively. Besides, the randomly selected attack targets are the XOR of the input and output of the 9^{th} S-box in the last round of AES-128 and the output of the 16^{th} S-box in the last round of the encryption algorithm, respectively. The other classical SCA attack, namely, correlation analysis attack (CAA) [17, 21, 22], is also used for comparison. The security metric, guess entropy [23], serves as the evaluation metric of the key recovery attacks. The attack results depicted in Figure 8 confirm the efficacy and efficiency of ANOVA_SCA. In particular, ANOVA_SCA is even much more efficient than CAA for the masked implementation.

We also estimate the average runtimes and the average memories when performing ANOVA_SCA and CAA, respectively. These two attacks are repeated 10 times with 30,000 power traces acquired from the FPGA implementation in Section A. This experiment is executed on the same HP laptop and MATLAB software platform mentioned in Section C. The statistical results are plotted in Figure 9. It can be viewed that it is feasible to apply the proposed attack for security assessment because of its practical utility and high efficiency.

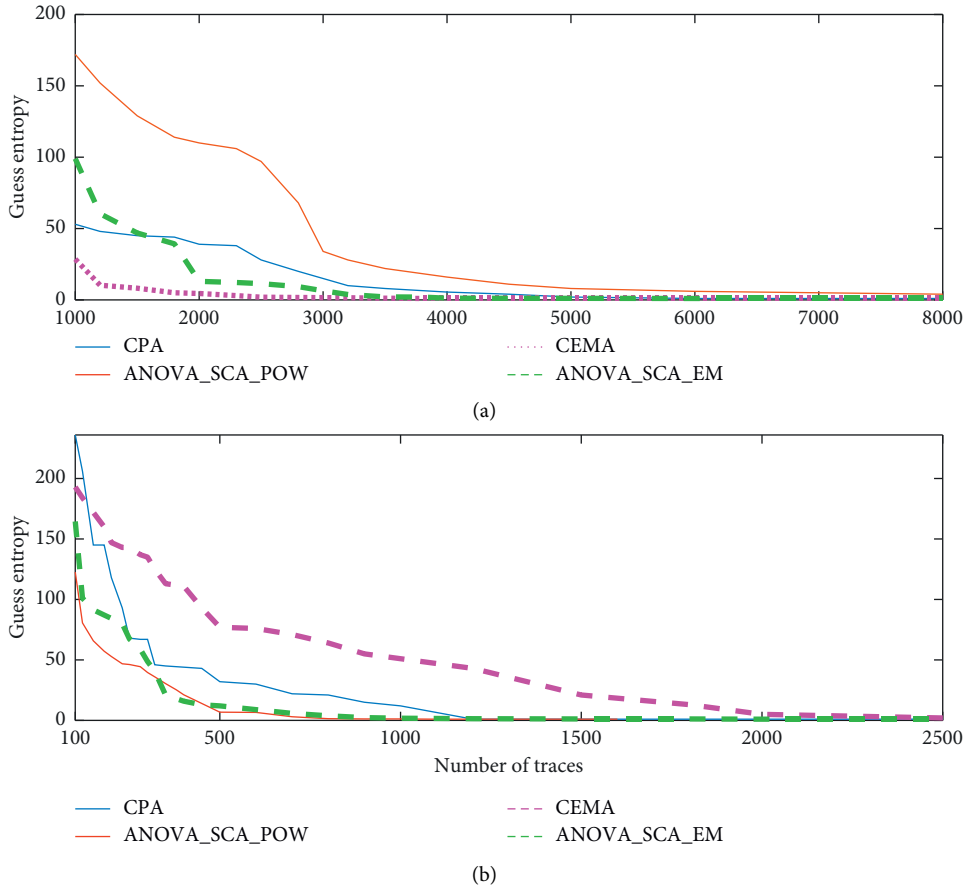


Figure 8: Guess entropy curves of CAA and ANOVA_SCA for the FPGA implementation and the 8-bit MCU implementation. Upper row: FPGA implementation. Lower row: MCU implementation. "CPA": CAA attack exploiting power traces. "ANOVA_SCA_POW": ANOVA_SCA attack exploiting power traces. "CEMA": CAA attack exploiting EM traces. "ANOVA_SCA_EM": ANOVA_SCA attack exploiting with EM traces.

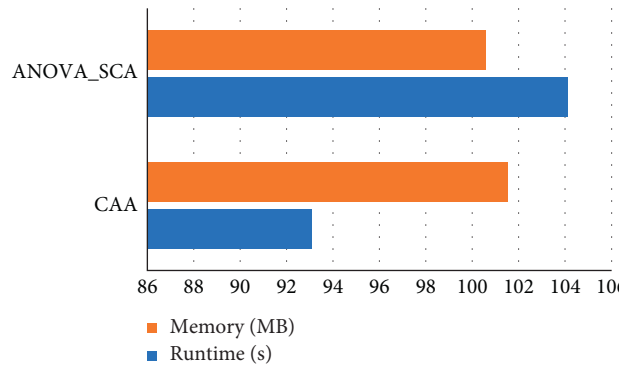


Figure 9: Runtimes and memories of ANOVA_SCA and CAA.

5. Conclusions

The paper develops a black-box side-channel leakage detection approach, which is designed for the security assessment of cryptographic devices. The proposed detection method is based on the one-way ANOVA, which ensures that the evaluation is highly efficient—in terms of both time and memory. On the basis of monochannel leakage detection, the proposed method is extended to detect multichannel leakages as well. Compared with utilizing monochannel leakage individually, the use of the fused leakage is beneficial in finding much more leakage points efficiently. Furthermore, the proposed detection method is capable of applying for the key recovery attack, when a key-

dependent immediate variable serves as the input of the method. In the future, the extension of our proposed approach to enhance the performance of multichannel leakage detection will be investigated further.

Data Availability

The side-channel measurement data and codes used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (nos. 61802186, 61472189, 61572255, and 62002167).

References

[1] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *Proceedings of the CRYPTO 1996*, pp. 104–113, Santa Barbara, CA, USA, August 1996.

[2] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proceedngs of the CRYPTO 1999*, pp. 388–397, Santa Barbara, CA, USA, August 1999.

[3] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: concrete results," in *Proceedings of the Cryptographic Hardware and Embedded Systems -2001*, pp. 251–261, Paris, France, May 2001.

[4] D. Genkin, A. Shamir, and E. Tromer, "RSA key extraction via low-bandwidth acoustic cryptanalysis," in *Proceedings of the CRYPTO 2014*, pp. 444–461, Santa Barbara, CA, USA, August 2014.

[5] G. Goodwill, B. Jun, J. Jaffe et al., "A testing methodology for side-channel resistance validation," in *Proceedings of the NIST NIAT 2011*, pp. 115–136, Nara, Japan, September 2011.

[6] S. Bhasin, J. Danger, S. Guilley et al., "NICV: normalized interclass variance for detection of side-channel leakage," in *Proceedings of the EMC'14*, pp. 310–313, Tokyo, Japan, May 2014.

[7] J. Liu, Z. Guo, D. Gu et al., "Enhanced side-channel leakage detection method by considering combinational logic," *China Communications*, vol. 12, no. 6, pp. 1–10, 2015.

[8] L. Mather, E. Oswald, J. Bandenburg et al., "Does my device leak information? an a priori statistical power analysis of leakage detection tests," in *Proceedings of the ASIACRYPT 2013*, pp. 486–505, Bengaluru, India, December 2013.

[9] F. Durvaux and F. Standaert, "From improved leakage detection to the detection of points of interests in leakage traces," in *Proceedings of the EUROCRYPT 2016*, pp. 240–262, Vienna, Austria, May 2016.

[10] A. A. Ding, C. Chen, and T. Eisenbarth, "Simpler, faster, and more robust *t*-test based leakage detection," in *Proceedings of the COSADE 2016*, pp. 163–183, Graz, Austria, April 2016.

[11] A. Moradi, B. Richter, T. Schneider et al., "Leakage detection with the χ^2-test," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 1, pp. 209–237, 2018.

[12] W. Yang, H. Zhang, Y. Gao et al., "Side-Channel leakage detection based on constant parameter channel model," in *Proceedings of the International Conference on Computer Design 2020*, pp. 553–560, Hartford, CT, USA, October 2020.

[13] G. Becker, J. Cooper, E. DeMulder, G. Goodwill et al., "Test vector leakage assessment (TVLA) methodology in practice," in *Proceedings of the International Catholic Migration Commission 2013*, Gaithersburg, MD, USA, September 2013.

[14] D. B. Roy, S. Bhasin, S. Guilley, A. Heuser, S. Patranabis, and D. Mukhopadhyay, "CC meets FIPS: a hybrid test methodology for first order side channel analysis," *IEEE Transactions on Computers*, vol. 68, no. 3, pp. 347–361, 2019.

[15] S. McKillup, *Statistics Explained: An Introductory Guide for Life Scientists*, Cambridge University Press, Cambridge, UK, 2nd edition, 2011.

[16] W. Snedecor and C. George, *Statistical Methods*, Iowa State University Press, Sioux, IA, USA, 8th edition, 1989.

[17] W. Yang, Y. Zhou, Y. Cao, H. Zhang, Q. Zhang, and H. Wang, "multi-channel fusion attacks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1757–1771, 2017.

[18] T. Schneider and A. Moradi, "Leakage assessment methodology–a clear roadmap for side-channel evaluations," in *Proceedings of the Cryptographic Hardware and Embedded Systems - 2015*, pp. 495–513, SaintMalo, France, September 2015.

[19] P. Bottinelli and J. W. Bos, "Computational aspects of correlation power analysis," *Journal of Cryptographic Engineering*, vol. 7, no. 3, pp. 167–181, 2017.

[20] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, Springer, New York, NY, USA, 2007.

[21] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Proceedings of the Cryptographic Hardware and Embedded Systems-2004*, pp. 16–29, Boston, MA, USA, August 2004.

[22] E. Peeters, *Advanced DPA Theory and Practice*, Springer, New York, NY, USA, 2013.

[23] F.-X. Standaert, T. G. Malkin, and M. Yung, "A unified framework for the analysis of side-channel key recovery attacks," in *Proceedings of the EUROCRYPT 2009*, pp. 443–461, Cologne, Germany, April 2009.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6695858, 10 pages
https://doi.org/10.1155/2021/6695858

Research Article

Calibrating Network Traffic with One-Dimensional Convolutional Neural Network with Autoencoder and Independent Recurrent Neural Network for Mobile Malware Detection

Songjie Wei, Zedong Zhang, Shasha Li, and Pengfei Jiang

School of Computer Science & Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Correspondence should be addressed to Songjie Wei; swei@njust.edu.cn

Received 30 December 2020; Revised 3 February 2021; Accepted 20 February 2021; Published 27 February 2021

Academic Editor: Liguo Zhang

In response to the surging challenge in the number and types of mobile malware targeting smart devices and their sophistication in malicious behavior camouflage, we propose to compose a traffic behavior modeling method based on one-dimensional convolutional neural network with autoencoder and independent recurrent neural network (1DCAE-IndRNN) for mobile malware detection. The design solves the problem that most existing approaches for mobile malware traffic detection struggle with capturing the network traffic dynamics and the sequential characteristics of anomalies in the traffic. We reconstruct and apply the one-dimensional convolutional neural network to extract local features from multiple network flows. The autoencoder is applied to digest the principal traffic features from the neural network and is integrated into the independent recurrent neural network construction to highlight the sequential relationship between the highly significant features. In addition, the *Softmax* function with the *LReLU* activation function is adjusted and embedded to the neurons of the independent recurrent neural network to effectively alleviate the problem of unstable training. We conduct a series of experiments to evaluate the effectiveness of the proposed method and its performance for the 1DCAE-IndRNN-integrated detection procedure. The detection results of the public Android malware dataset CICAndMal2017 show that the proposed method achieves up to 98% detection accuracy and recall rates with clear advantages over other benchmark methods.

1. Introduction

With the rapid development of mobile cellular networks and the prevailing of smart devices, mobile applications have become more indispensable to people's routine life. Meanwhile, application users are confronted with increasingly serious security and privacy threats from mobile malware [1]. The coexistence of modern mobile devices with e-commerce, personal payment, social communication, and other related applications is so critical that we will never overestimate the importance of mobile application security insurance and malware detection.

Many mobile applications operate by interacting with other devices and remote servers on Internet connections. Particularly, when malware carries out malicious actions, it exposes obvious characteristics in network interaction traffic. Previous researches revealed that more than 90% of malware-infected mobile terminals are controlled by botnets controlling malware behavior through network or SMS instructions [2]. The discovery of such fact provides the possibility to explore malware detection methods based on network behavior characteristics as traffic dynamics. Malicious application detection based on network traffic dynamics is to recognize and model malware behaviors from the perspective of network traffic data, which has been recently thoroughly studied by both academia and industry. Malware detection systems based on traffic analysis can be deployed flexibly on network access points or clouds.

Malware detection method based on the statistical characteristics of network flows has the advantages of low

computation cost and avoidance in deep packet inspection and is applicable for both plain and encrypted traffic with privacy protection. However, mobile Internet is a very dynamic and complex network system, and its traffic statistical patterns demonstrate nonstationary, nonlinear, and super chaotic characteristics. When the network application environment or network scale varies, it is difficult to accurately and faithfully reflect the behavior of network traffic based on the characteristics of a single network flow. This is mainly because the accumulated information in network traffic may also change accordingly. Malware network behavior is usually a sequence of continuous behaviors. That is, when a malicious behavior appears in the flow from a source IP, the probability that the following behaviors continue the maliciousness is high, meaning that a network communication behavior has sequential forward and backward relevance. Therefore, most malicious traffic events follow a time-series pattern, and current network traffic can be classified according to the observed past network traffic records [3]. Previous experimental results also show that the sequential characteristics of network flow as one of the important feature dimensions in the traffic detection problem can effectively improve the performance of malicious traffic detection [4, 5].

This paper combines the strength of deep learning models and mobile malware traffic analysis with contributions in the following aspects:

(1) We first propose a mobile malware traffic detection method based on one-dimensional convolutional neural network with autoencoder and independent recurrent neural network (1DCAE-IndRNN) to capture the traffic dynamics and key features of mobile malware's Internet interaction. 1DCAE (one-dimensional convolutional neural network with autoencoder) is adapted to extract the local features of network flow data to ensure its temporal variability and sequence correlation, and the autoencoder helps recognize and concentrate on the efficient features in the traffic.

(2) Then IndRNN (independently recurrent neural network) is designed with fine-tuning to obtain the sequential correlations between high-level sequential-temporal features in traffic flows. It can digest valuable information for malware traffic detection. IndRNN is a revised type of recurrent neural network, which learns the long-term dependence of time pattern in large-scale sequence data. We add the *Softmax* function to IndRNN and use the *LReLU* function as an activation function, which alleviates the instability of IndRNN training to some extent.

(3) We conduct evaluation experiments on publicly available malware dataset and benchmark the proposed method with other typical detection approaches in the literature. The experimental results on the public Android malware dataset CICAndMal2017 show that the method proposed in this paper has higher detection accuracy and recall rate, which are superior to the traditional machine learning methods and the recent deep neural network models by others.

The rest of the paper is organized as follows. Section 2 surveys the recent work in related areas. Section 3 briefly introduces IndRNN. Section 4 explains the structure of the 1DCAE-IndRNN model proposed in this paper and the improvements to IndRNN. Section 5 presents the experimental results and analysis. At last, Section 6 concludes with further insight and discussion.

2. Related Work

Mobile malware detection is a critical part for building a mobile network defense system [6]. Various theoretical and practical methods for mobile malware detection are emerging recently. Dhaya and Poongodi [7] proposed a malware detection system based on N-gram algorithm. In their system, the behavioral characteristics of software are first obtained through static analysis. Then the N-gram algorithm is used for classification and identification, so that the category of the software is either benign or malignant. Khatri and Abendroth [8] developed a mobile network-based malware detection system to identify malicious activities in network and protect end users from mobile malware attacks. Nguyen and Yoo [9] built a malicious software detection system for SDN mobile devices based on network behaviors. The system consists of three algorithms: IP blacklist, connection success rate, and connection rate. It develops malicious software detection methods in a more effective and flexible way. However, the fundamental problem of these related works is that they cannot get rid of their dependence on feature engineering; that is, the quality of malware detection methods proposed depends largely on the quality of their feature extraction.

In order to alleviate the dependency problem on feature engineering, mobile malware detection methods based on deep learning have gradually attracted the attention of academia. Amalina et al. [10] proposed a solution for malware detection using machine learning to evaluate various network traffic characteristics. Shabtai et al. [11] proposed a new framework for Android malware detection, first monitoring features from Android devices and then using machine learning classifiers to determine whether the software is malware. Wang et al. [12] proposed a malicious software traffic classification method. The first 784 bytes of single data packets is transformed into a two-dimensional gray-scale image, and then the image is recognized by a convolutional neural network. This method does not need the manually extracted features but directly uses the original flow as input data for the classifier. They conducted experiments based on the self-created traffic dataset USTC-TFC2016, using three classifiers in two cases, and the final average accuracy can reach 99.41%. However, they use both packet header information and payload as evidence for classification, which are limitedly available in a dataset collected in the local

network. In a real network beyond LAN, packet header may change when traversing the network, which challenges the reuse of a previously trained model everywhere [13].

Obviously, an ideal traffic analysis and detection model for malware traffic behaviors is desirable without dependency on packet-content inspection or network environment restriction. While the monitored raw traffic from network is temporal and sequential with features available for extraction on different protocol layers within different time windows and while such features are all mixed up with possible obfuscation and anonymization, the underlying behavior dynamics originated from the application's designed functional and network interaction casualty can never be concealed or anonymized. Instead of recognizing malware by traffic feature appearance, this paper explores the possibility to reconstruct and reorganize the available features to build their interconnection spanning on the time scale, that is, the temporal-sequential correlations and their underlying application network dynamics.

3. IndRNN Basics

IndRNN evolves as a new recurrent neural network model by Li et al. [14]. IndRNN can effectively solve the problem of gradient disappearance and explosion by adjusting the time-based gradient backpropagation and can retain long-term memory and process long sequences. The definition of an IndRNN is

$$h_t = \sigma\left(W_{x_t} + \mu \odot h_{t-1} + b\right), \tag{1}$$

where $x_t \in \mathbb{R}^M$ and $h_t \in \mathbb{R}^N$ represent the input and hidden layer states of IndRNN at time t. $W_t \in \mathbb{R}^{M\times N}$ and $\mu \in \mathbb{R}^N$ represent the weights in the corresponding state, and $b \in \mathbb{R}^N$ is the neuron's bias terms. N and M are the number of neurons in the current layer and the length of the input sequence. σ represents the activation function of the neurons, and $\odot$ is the Hadamard product. For the n-th neuron at time t, the hidden state $h_{n,t}$ can be obtained by the following formula:

$$h_{n,t} = \sigma\left(W_n X_t + \mu_n \odot h_{n,t-1} + b_n\right), \tag{2}$$

where μ_n is the n-th neuron weight of the layer and $h_{n,t-1}$ stands for the previous hidden state. A neuron only receives information from the input and its own hidden state in the previous time step. Each neuron in layers is independent of others and independently processes a type of space-time pattern. The internal structure of IndRNN is shown in Figure 1.

Here, "Weight" represents the input weight processing, and "Recurrent" represents the probability of calculating the weight of each recurrent. "Recurrent + *ReLU*" means a recurrent process with *ReLU* (Rectified Linear Unit) as the activation function in each step. The *ReLU* activation function can reduce the disappearance of gradients between layers, so that multiple layers of IndRNN neural networks can be efficiently stacked. "Batchnorm" stands for batch normalization and is used before or after the activation function.

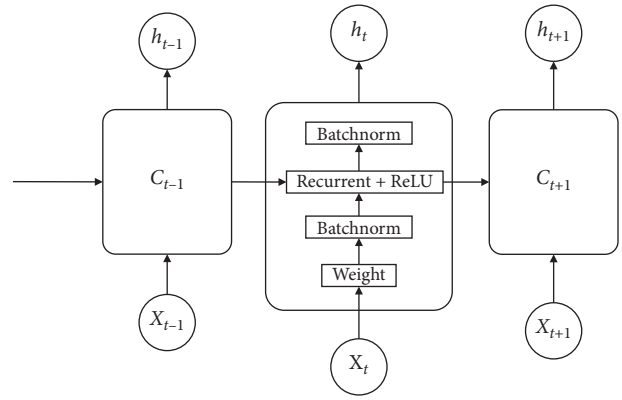


Figure 1: Structure diagram of independent recurrent neural network.

Since IndRNN can solve the problem of gradient explosion and disappearance over time, gradients are efficiently propagated between different steps. Therefore, IndRNN is usually deeper and longer than traditional RNN. IndRNN uses the *ReLU* function as the activation function, and the length of the memory obtained by learning different features is not the same. This greatly enhances the interpretability of the model, and the robustness of the model after training is also improved.

4. 1DCAE-IndRNN Model Design and Improvement

Network traffic is essentially sequential data with temporal features, being observed as a single one-dimensional byte flow organized in a hierarchical structure. Among the flow, bytes, packets, sessions, and their combinations are analogous to characters, words, sentences, and the whole article in natural language processing. A typical task of natural language classification usually extracts two levels of features. One is the features between words in a single sentence, which can identify the semantics expressed in the sentence by learning such features. The other is the relationship features between different sentences in a paragraph and then the overall semantics of the whole paragraph in the context. Inspired by this idea, we propose to construct a 1DCAE-IndRNN model to learn local and sequential features of network flows. 1DCAE can extract short-time and frequency-domain features from time-domain signals. It has achieved great success in the field of audio generation and machine translation, being very effective for simple tasks such as sequence classification and time-series prediction and with a faster training speed. IndRNN can learn the traffic sequence information of several previous time points and feed it back to the traffic state of the current time point to realize the learning of traffic sequence characteristics.

The 1DCAE-IndRNN malware traffic detection model, which can learn the correlation and local characteristics between network flow sequences, has a multilayer system structure, including separate input layer, hidden layers, and output layer. The overall 1DCAE-IndRNN network structure designed is shown in Figure 2. After inputting

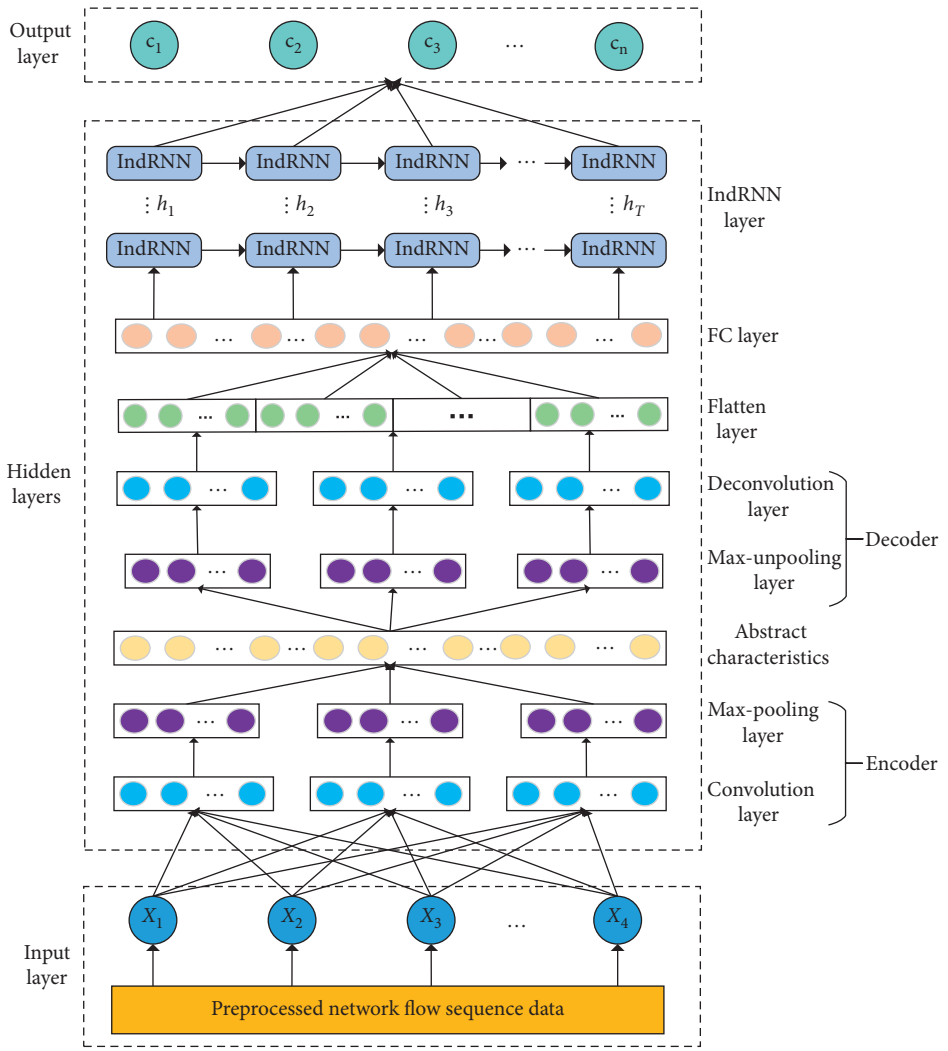


Figure 2: 1DCAE-IndRNN model structure diagram for malware traffic detection.

the traffic data with a sequential relationship into the model, the output layer finally gets the prediction result using the first T time points of data after the hidden layer calculation.

4.1. Input Layer. The input layer is responsible for importing data, which is the network flow features after traffic preprocessing. The 1DCAE-IndRNN input layer requires that the input data is three-dimensional, containing batch size, input dimension, and time steps. Batch size refers to the number of samples fed into the neural network at one time. The time step represents the time point length of the model memory. The input dimension represents the length inherent in a single time dimension of a single sequence of input data.

4.2. Hidden Layers. The hidden layers are the core part of learning and correlating the local and sequential features in network flow data. It consists of 1DCAE layer with the decoder (including convolution layer; and maxpooling layer), the encoder (including max-unpooling layer and deconvolution layer), flatten layer, IndRNN layer, and fully connected layer. Each IndRNN layer contains multiple IndRNN neuron nodes.

1DCNN mainly implements the function of local feature extraction by stacking convolution layer and pooling layer. The convolutional layer can be trained to obtain a set of optimal convolution kernels that meet the minimum loss function and use the convolution kernel to implement automatic feature extraction. The pooling layer can extract the most important features from the convolution layer and reduce the dimension in time dimension. The one-dimensional convolution kernel is equivalent to the observation window on the time axis of the network flow sequence. The convolution operation can extract short-term features on the time axis. The pooling layer further aggregates these short-term features and retains some of them. After the operations in the one-dimensional convolution layer and the pooling layer, the local characteristics of the network flow sequence are well analyzed and retained. The entire 1DCNN workflow is shown in Figure 3. The red and blue rectangles in the figure represent one-dimensional convolution kernels.

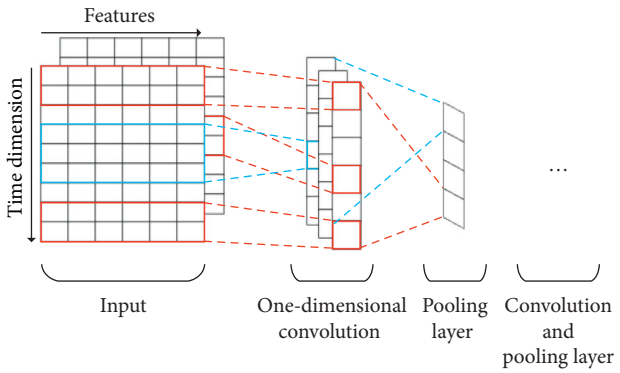


Figure 3: Workflow of one-dimensional convolutional layer.

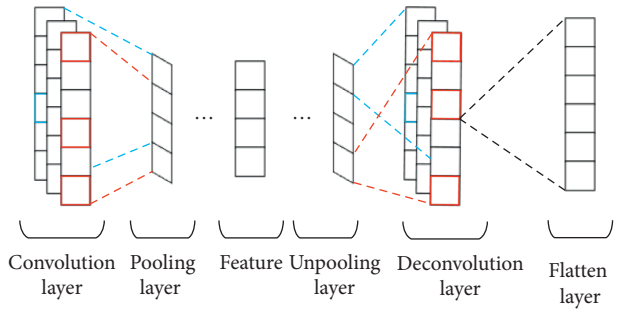


Figure 4: Workflow of encoder and decoder layer.

The encoder performs a convolution operation on the input layer data to extract local features based on 1DCNN. The decoder is to perform a deconvolution operation on the hidden layer data, so that the output data are restored to the dimension of the input data. Then the neural network minimizes the error between input and output by iterating repeatedly. The entire encoder workflow is shown in Figure 4.

Flatten layer is used for the transition from the 1DCAE layer to the IndRNN layer; that is, multidimensional input is one-dimensionalized. After the 1DCAE layer, a fully connected layer is needed to map the high-dimensional features to the low-dimensional ones and retain the useful information.

In IndRNN, we use the *ReLU* activation function to easily implement efficient stacking of multilayer IndRNN networks. However, since the *ReLU* function can output 0, the whole history information of neurons may be discarded and has to be relearned from the current time. Therefore, the stability of IndRNN in the training process is insufficient, and the prediction results are very unstable. Due to the temporal and dynamic characteristics of network traffic, it is difficult for the model to learn the rules and capture useful mutation information.

To alleviate the above problem, we put the recurrent weight information of IndRNN into the *Softmax* function and replaced the *ReLU* function with the *LReLU* function. The general meaning of the *Softmax* function is to normalize the vector, highlighting the maximum value and suppressing other components far below the maximum value. For the n-th neuron at time t, the hidden state $h_{n,t}$ becomes

$$h_{n,t} = \sigma\left(W_n X_t + \frac{e^{\mu_n \odot h_{n,t-1}}}{\sum_{i=1}^{T} e^{\mu_n \odot h_{n,t-1}}} + b_n\right). \tag{3}$$

Softmax function calculates the probability of each repeated weight, which affects the size of its loss function and ensures the effective propagation of gradient information. The *LReLU* function has a small slope for the negative value input, and the calculation method is shown in equation (4).

$$y_i = \begin{cases} x_i, & x_i \geq 0, \\ a_i x_i, & x_i < 0, \end{cases} \tag{4}$$

where a_i is a fixed parameter in the (0, 1). Since the derivative of the *LReLU* function is always not zero, it can reduce the occurrence of "dead" neurons, which effectively solves the problem that neurons do not learn when the *ReLU* function enters the negative range. In this way, it can be determined which weight in the recursive weight matrix has a larger probability value. If the weighted probability is large, the weighted probability is retained, otherwise suppressed. The improved IndRNN not only evaluates the impact of the previous time step on the current time step with more accuracy but also reduces the training complexity of the model. The internal structure of the improved IndRNN is shown in Figure 5.

4.3. Output Layer. The output layer consists of a fully connected layer, and the input is the output of the last hidden layer. The number of nodes in the output layer is the number of categories in the training set. After the multiclassification function *Softmax* is processed, the final classification result is obtained.

5. Experiments and Evaluation

5.1. Experimental Environment. All the following experiments are conducted on a desktop with 16 GB of RAM, Intel Core i7-7700 3.6 GHz CPU, and Nvidia GeForce GTX 1660Ti graphics card. Python 3.5 is used for programming. 1DCAE-IndRNN model is built on Keras (2.1.0), an open-source deep learning framework with TensorFlow (1.12.0) as the back end, with GPU acceleration. Benchmark algorithms are implemented based on the machine learning framework scikit-learn (0.18.0) and the deep learning ones on Keras.

5.2. Dataset Introduction and Preprocessing. CICAndMal2017 [15] is a new Android malware dataset proposed by the Canadian Institute for Cybersecurity. The collectors ran both the malware and benign applications on real smartphones to avoid runtime behavior modification of advanced malware samples that can detect the emulator environment. They installed 5,000 of the collected samples (426 malware and 5,065 benign) on real devices. The dataset includes benign and malicious application samples, which are divided into four malware categories, including adware, ransomware, scareware, and SMS malware. Malicious samples come from 42 unique malware families.

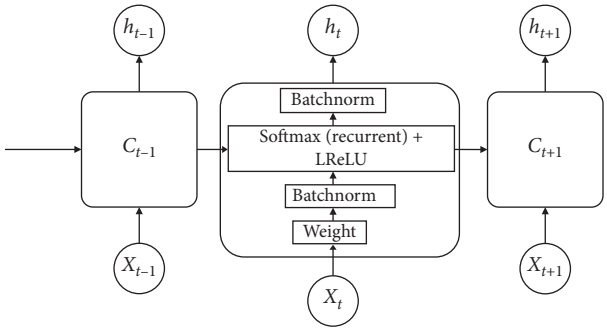


Figure 5: Internal structure of the improved IndRNN.

CICAndMal2017 dataset is fully labeled and includes network traffic, logs, API/SYS calls, phone statistics, and memory dumps of 42 malware families, with 84 features extracted during all the three execution states (installation, before restart, and after restart). The dataset contains more than 1.8 million pieces of data, covering most of the dynamic features of Android software. This dataset is suitable for validating the effectiveness of dynamics-based malware detection. In order to accurately evaluate the proposed 1DCAE-IndRNN's ability to recognize mobile malware traffic, this paper conducts experiments as binary classification tasks of benign and malicious applications.

In order to facilitate the calculation and analysis of data by the model, we perform the following preprocessing steps on the CICAndMal2017 dataset. Firstly, we exclude nonnumeric features, including Flow ID, Source IP, Source Port, Destination IP, Destination Port, and Timestamp. These nonnumeric features cannot be quantitatively modeled by the neural network and are irrelevant to maliciousness. Secondly, we select all the statistical characteristics of network traffic, totaling 68, some of which are shown in Table 1. We use 0 to fill in missing value fields. Finally, we use the min–max normalization method to reshape the data. The linear transformation of the original data by min–max normalization makes the result fall in 0~1. One-third of the total data is used as the test set and the rest for training.

5.3. Malware Detection Experiment and Analysis. In order to train a 1DCAE-IndRNN model structure with better performance, this experiment builds a one-dimensional convolutional layer containing 1 to 3 layers and an IndRNN layer containing 1 to 4 layers based on the model designed in Section 4. There are 12 different 1DCAE-IndRNN models. The filling mode used on the boundary of the one-dimensional convolution layer is "valid," and the sliding steps are all 1. The number of neuron nodes in the fully connected layer between the 1DCAE layer and the IndRNN layer is set to 64. The slope of the *LReLU* function in the negative interval is set to 0.05. The specific structural parameters of the 1DCAE layer and the IndRNN layer are shown in Tables 2 and 3.

Here, Conv1D (x, y) and deConv1D (x, y) indicate that the convolution kernel size is x and the number of convolution kernels is y. Maxpooling (2) and unMaxpooling (2) represent the max-pooling layer window size of 2. IndRNN (n) indicates that there are n IndRNN neuron nodes.

Although the robustness of IndRNN after training is stably good, due to the strong learning ability of the neural network model itself, overfitting may still occur. Therefore, a dropout layer is added after each IndRNN layer to prevent overfitting, and the neuron dropout rate of the dropout layer is set to 0.5.

In order to reduce the computational cost and randomness of model training, our experiment uses the minibatch training method, and the batch size of this experiment is set to 128. The training loss function of the model is the cross-entropy loss function. The optimizer uses the Adam method. The initial learning rate is 0.0005 and the number of training rounds is 150. Due to the small scale of the datasets, this experiment configures the time step of 1DCAE-IndRNN to 10, where 1DCAE-IndRNN can remember the traffic sequence information of the previous 10 time periods. The model extracts the local features of network flow data through 1DCAE layer, uses IndRNN layer to learn the sequential association information between high-level features, and finally outputs the prediction category through a fully connected layer.

The designed 1DCAE-IndRNN models with 12 different structures are each trained 3 times, and the average accuracy of each structure is obtained based on the results of the 3 experiments. The output results with different structures are shown in Table 4, where the structure indexes correspond to those in Tables 2 and 3.

From the experimental results in Table 4, we can notice that the classification accuracy of 1DCAE-IndRNN models with different structures has some differences. The accuracy of most models is more than 90%. When the number of 1DCAE layers is 2 and the number of IndRNN layers is 3, the classification accuracy of the models is the highest, up to 98.32%. With a small 1DCAE layer, the classification performance of the model can be improved by increasing the number of IndRNN layers, although such increase may not continue infinitely. For example, when the number of 1DCAE layers is 2, with the number of IndRNN layers being 4, the classification accuracy of the model declines. But there is no significant decline, which is inseparable from the advantages and robustness of the gradient in the training of the IndRNN model. When the number of 1DCAE layers is 1, adding another 1DCAE layer, the classification accuracy of the model has been greatly improved, indicating that the 1DCAE layer is very effective in extracting local features of network flow sequence data, and two 1DCAE layers are suitable for the model. The experimental data in this paper have fewer feature dimensions. Complicating the model may not improve the detection performance of the model but will definitely increase the training and test time overhead for the model. Therefore, we select the optimal hidden layer structure (2 1DCAE layers; 3 IndRNN layers) and conducts 3 experiments again. Accuracy, recall, F1-score, and training time are calculated as evaluation metrics. The experimental results are shown in Table 5. It can be seen that the performance of the 1DCAE-IndRNN model is still very stable, and the training time is significantly elongated.

Table 1: Partially selected features.

Feature	Description
Fwd_Packet_Length_Std	Standard deviation of the size of packet in forward direction
Init_Win_Bytes_Forward	Total number of bytes sent in the initial window in the forward direction
Init_Win_Bytes_Backward	Total number of bytes sent in the initial window in the backward direction
Bwd_Packet_Length_Min	Minimum of the size of packet in backward direction
Total_Length_BwdPackets	Total size of the packet in backward direction
Flow_IAT_Max	Maximum interarrival time of the packet

Table 2: Structural parameters of different designs for 1DCAE layer.

Layers	Structural parameters
1	Conv1D (20, 8), maxpooling (2) unMaxpooling (2), deConv1D (20, 8)
2	Conv1D (20, 8), maxpooling (2), Conv1D (30, 4) deConv1D (30, 4), unMaxpooling (2), deConv1D (20, 8)
3	Conv1D (20, 8), maxpooling (2), Conv1D (30, 4), maxpooling (2) Conv1D (40, 2), maxpooling (2) unMaxpooling (2), deConv1D (40, 2), unMaxpooling (2), deConv1D (30, 4), unMaxpooling (2), deConv1D (20, 8)

Table 3: Structural parameters of different designs for IndRNN layer.

IndRNN layers	Structural parameters
1	IndRNN (20)
2	IndRNN (30), IndRNN (30)
3	IndRNN (30), IndRNN (40), IndRNN (30)
4	IndRNN (30), IndRNN (40), IndRNN (40), IndRNN (30)

Table 4: Test set accuracy of 1DCAE-IndRNN models with different hidden layer structures.

IndRNN1DCAE	1 (%)	2	3	4 (%)
1	86.21	87.16%	91.79%	91.23
2	91.32	95.74%	98.32%	97.32
3	93.29	96.48%	97.11%	95.23

Table 5: Experimental results of the optimal 1DCAE-IndRNN model.

Number	Accuracy (%)	Recall (%)	F1-score	Time (min)
1	98.32	98.32	98.31	43.6
2	98.27	98.26	98.26	41.5
3	98.29	98.27	98.27	42.4

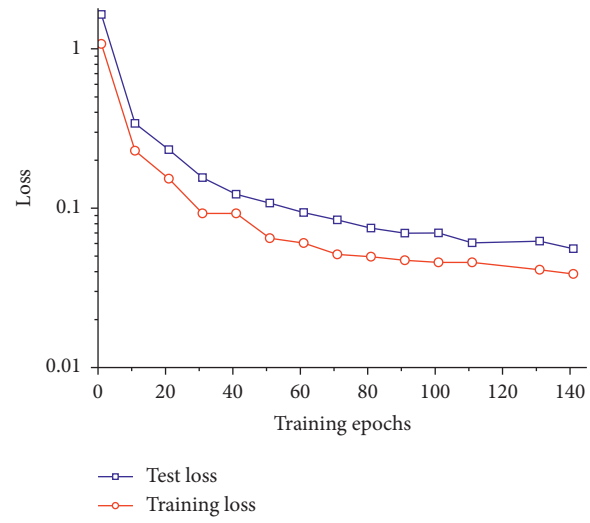


Figure 6: Optimal 1DCAE-IndRNN model loss value curves.

Figure 6 shows the loss curve of the optimal 1DCAE-IndRNN model in training and test. It can be seen that, in the first 40 rounds of training, the loss value of the model decreases rapidly, which shows that the convergence speed of the model is fast. After 80 rounds of training, the training set loss of the model has become stable. During the whole training process, the training loss value and test loss value of the model do not fluctuate greatly, indicating that the training stability of the model is ensured. The improvement of IndRNN effectively alleviates the problem of model training instability. Therefore, the 1DCAE-IndRNN model proposed in this paper is superior in terms of convergence speed, training stability, and classification performance.

In addition, we investigate the influence of the number of training data on the classification performance of the 1DCAE-IndRNN model. We use different numbers of training sets to train the model, and each trained model is evaluated on the complete test set. It is worth mentioning

that the proportion of various types of traffic in the training set is unchanged. The value range of the training data ratio is [0.1, 0.2, 0.4, 0.6, 0.8, 1.0]. Figure 7 shows the test set accuracy curve when training with different scale training sets.

According to the experimental results in Figure 6, when the proportion of training data is only 0.1, the accuracy of the test set is low due to insufficient features extracted by the model, and the fluctuation is obvious during training, and the convergence speed is also slow. When the amount of training data is continuously increased, more and more features are extracted by the model, where the final classification accuracy of the model is rising, and the convergence speed is continuously accelerated with a more stable training process. Therefore, although smaller training dataset can achieve 92.89% accuracy with less training time, the training process is prone to instability, and the model convergence speed is slow. More training data tends to make the detection performance of the model better with a slight increase in training time. Under the premise of ensuring the accuracy of model detection, the amount of training data needs to be controlled within a certain range to obtain the most suitable training time and classification performance.

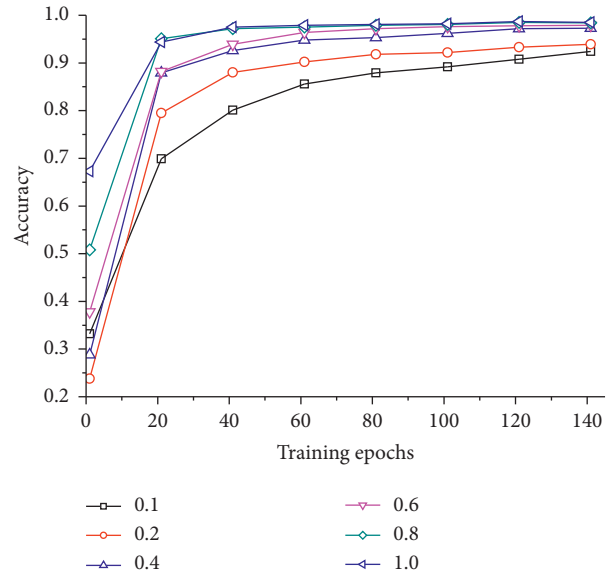


Figure 7: Test set accuracy curves with different training set sizes.

5.4. Comparison with Other Methods. In order to verify the advantages of the 1DCAE-IndRNN method proposed in this paper for malware traffic detection, two classical machine learning methods and three deep learning methods are selected from the literature for experimental comparison. Classical machine learning methods include random forest (RF) [16] and XGBoost [17]. Deep learning methods include deep neural networks (DNN) [18], recurrent neural networks (RNN) [19], and long short-term memory (LSTM) [20]. The application of these benchmark methods is briefly surveyed in the related work section. The time steps for LSTM and RNN are set to 10. Figure 8 shows the classification accuracy, recall rate, and F1-score of the five comparison classification models and 1DCAE-IndRNN on the test set.

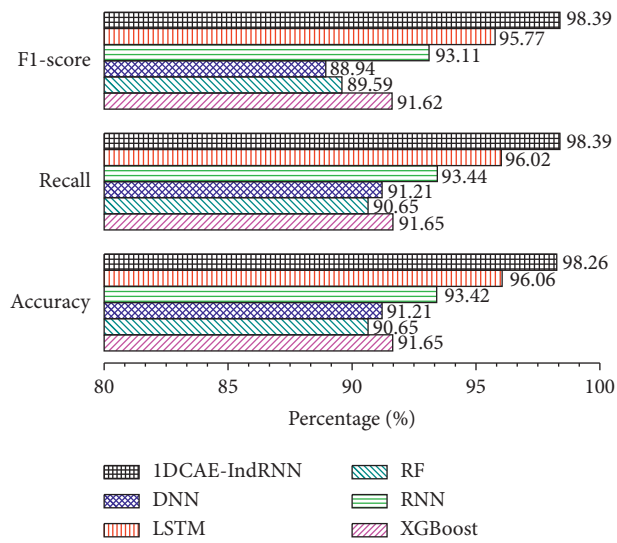


Figure 8: Comparison of classification performance with other methods.

According to the experimental results in Figure 8, the classification performance of the 1DCAE-IndRNN-based malware traffic detection method proposed in this paper is superior to all the comparison algorithms. Because XGBoost, DNN, and random forest models do not have the time memory function, they cannot take advantage of the sequential characteristics between network flows. The classification accuracy of these three models is lower than that of RNN and LSTM, and the F1-score is also poorer. RNN and LSTM are both models utilizing the time sequence characteristics between network flows, and the classification accuracies reach over 93%. However, they do not have 1DCAE to extract the local feature information of network flow sequence. Moreover, when the time step of RNN is a little longer, information loss is likely to occur. Therefore, the classification performance of RNN and LSTM is lower than that of the 1DCAE-IndRNN model.

For a real network environment, the time efficiency of the applied malware traffic detection model is not ignorable. The shorter the online identification execution time is, the better it is to take defensive measures in real time. The shorter the model training time is, the faster it is to update the model parameters. Therefore, this paper compares the training time of a round of deep learning model (DNN, RNN, LSTM, and 1DCAE-IndRNN) on the same dataset and the test time of single data after the model training. The experimental results are concluded in Figures 9 and 10, respectively. It is worth mentioning that these deep learning models use the same GPU acceleration hardware during training.

Since the convolution and pooling operation of 1DCAE greatly reduces the number of characteristic parameters and the number of IndRNN parameters is relatively small, the one-round training time of the 1DCAE-IndRNN model proposed is 31 seconds, and the test time of single data on the trained model is only 74 milliseconds, showing high training and detection efficiency. DNN does need to learn the

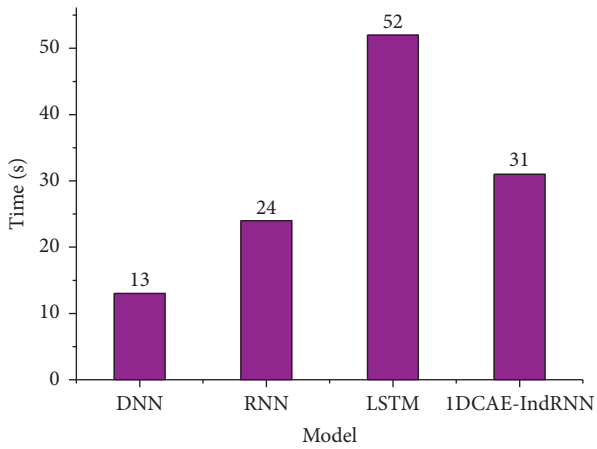


Figure 9: Single round training time of different neural network models.

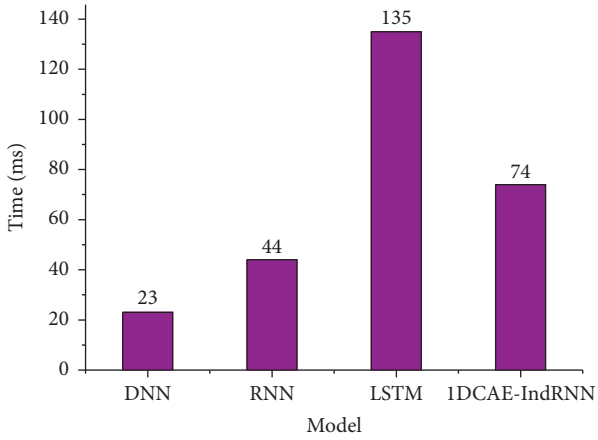


Figure 10: Test time of single data with different neural network models.

sequential information of multiple network flows, so the single round training time and single data test time are the smallest. Since the LSTM needs three gates to control the learning of the model, the parameters of the model itself are very large, which leads to a longer time for the parameter updating in the LSTM model. The number of parameters in RNN is less than that in the 1DCAE-IndRNN model, so the training time and test time of a single round are proportionally lower than the 1DCAE-IndRNN model.

6. Conclusions

This paper proposes a mobile malware traffic detection method based on 1DCAE-IndRNN, which aims to solve the problem that the current mobile malware traffic detection algorithm is struggling with capturing the dynamic changes and key temporally local information in abnormal traffic flows from mobile applications. In our design of 1DCAE-IndRNN, 1DCAE can extract local features of multiple network flows, and the independent recurrent neural network obtains the sequential relations between high-level features. In addition, activation functions are added to the IndRNN neurons to effectively alleviate the instability in the training procedure. We utilize the CICAndMal2017 dataset to conduct a series of simulation experiments and study the impact of the model structure variation and the training data volume difference on the detection performance. Experimental results show that the method proposed in this paper has higher detection accuracy, recall, and F1-score, which is superior to the common classic machine learning methods and the recently proposed deep neural network models. We demonstrate that, by capturing and characterizing the dynamic features of network traffic from the malicious application software, it is promising to differentiate the network behaviors between benign and malicious applications. And behavior-based classification has more insight into malware maliciousness than code appearance. In future work, we plan to further investigate how to generalize such network traffic modeling on behavior dynamics on malware family level. That means using the proposed method to not only separate benign from malware but also cluster and group those malicious ones for code heritage and attack originality.

Data Availability

CICAndMal2017 is a new Android malware dataset provided by the Canadian Institute for Cybersecurity. Lashkari, Arash Habibi et al. "Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification." 2018 International Carnahan Conference on Security Technology (ICCST). IEEE, 2018.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants nos. 61802186 and 61472189.

References

[1] D. He, S. Chan, and M. Guizani, "Mobile application security: malware threats and defenses," *IEEE Wireless Communications*, vol. 22, no. 1, pp. 138–144, 2015.

[2] Y. Zhou and X. Jiang, "Dissecting android malware: characterization and evolution," in *Proceedings of IEEE Symposium on Security and Privacy Conference*, vol. 95-109, San Francisco, CA, USA, May 2012.

[3] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *Proceedings of International Conference on Advances in Computing, Communications and Informatics*, vol. 1222-1228, IEEE, Udupi, India, September 2017.

[4] W. Wang, Y. Sheng, J. Wang et al., "HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2017.

[5] Z. Zou, J. Ge, H. Zheng et al., "Encrypted traffic classification with A convolutional long short-term memory neural network," in *Proceedings of International Conference on High*

Performance Computing and Communications, IEEE, Fiji, Oceania, December 2018.

[6] M. Mimura and H. Tanaka, "Long-term performance of a generic intrusion detection method using Doc2vec," in *Proceedings of International Symposium on Computing & Networking*, IEEE, Silicon Valley, CA, USA, January 2017.

[7] R. Dhaya and M. Poongodi, "Detecting software vulnerabilities in android using static analysis," in *Proceedings of International Conference on Advanced Communications, Control and Computing Technologies*, IEEE, Ramanathapuram, India, May 2014.

[8] V. Khatri and J.. Abendroth, "Mobile guard demo: network based malware detection," in *Proceeding of Trustcom/BigDataSE/ISPA*, IEEE, Washington, DC, USA, May 2015.

[9] T.-H. Nguyen and M. Yoo, "A behavior-based mobile malware detection model in software-defined networking," in *Proceedings of International Conference on Information Science and Communications Technologies*, Moscow, Russia, September 2017.

[10] F. Amalina, A. Feizollah, and N. Badrul Anuar, "Evaluation of machine learning classifiers for mobile malware detection," *Soft Computing*, vol. 20, no. 1, pp. 343–357, 2014.

[11] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "Andromaly": a behavioral malware detection framework for android devices," *Journal of Intelligent Information Systems*, vol. 38, no. 1, pp. 161–190, 2012.

[12] W. Wang, M. Zhu, X. Zeng et al., "Malware traffic classification using convolutional neural network for representation learning," in *Proceedings of International Conference on Information Networking*, IEEE, Shenzhen, China, March 2017.

[13] H.-k. Lim, J.-B. Kim, J.-S. Heo et al., "Packet-based network traffic classification using deep learning," in *Proceedings of International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, IEEE, Jeju Island, South Korea, April 2019.

[14] S. Li, W. Li, C. Cook et al., "Independently recurrent neural network (IndRNN): building A longer and deeper RNN," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[15] A. Lashkari, A. Kadir, L. Taheri et al., "Toward developing A systematic Approach to generate benchmark android malware datasets and classification," in *Proceedings of International Carnahan Conference on Security Technology (ICCST)*, IEEE, Montreal, Canada, October 2018.

[16] C. Wang, T. Xu, and X. Qin, "Network traffic classification with improved random forest," in *Proceedings of International Conference on Computational Intelligence and Security*, IEEE, Shenzhen, China, December 2015.

[17] Q. Chen, C. Guestrin, and X. Xgboost, "A scalable tree boosting system," in *Proceedings of ACM SIGKDD International Conference*, pp. 785–794, Anchorage, AK, USA, August 2016.

[18] W. Elmasry and A. Akhan Akbulut, "Halim zaimet, deep learning approaches for predictive masquerade detection," *Security and Communication Networks*, vol. 2018, Article ID 9327215, 24 pages, 2018.

[19] J. Du, V. Chi-Man, C. Philip Chen et al., "Novel efficient RNN and LSTM-like architectures: recurrent and gated broad learning systems and their applications for text classification," *IEEE Transactions on Cybernetics*, vol. 992 pages, 2020.

[20] C. H. Park and G.-H. Lee, "Jamming prediction for radar signals using machine learning methods," *Security and Communication Networks*, vol. 2020, Article ID 2151570, 9 pages, 2020.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6671132, 11 pages
https://doi.org/10.1155/2021/6671132

Research Article

A Novel Classified Ledger Framework for Data Flow Protection in AIoT Networks

Daoqi Han,[1] **Songqi Wu**,[1] **Zhuoer Hu**,[1] **Hui Gao**,[1] **Enjie Liu**,[2] **and Yueming Lu**[1]

[1]*Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China*
[2]*University of Bedfordshire, Institute for Research in Applicable Computing (IRAC), Luton, UK*

Correspondence should be addressed to Yueming Lu; ymlu@bupt.edu.cn

Received 26 December 2020; Revised 11 January 2021; Accepted 4 February 2021; Published 19 February 2021

Academic Editor: Liguo Zhang

The edge computing node plays an important role in the evolution of the artificial intelligence-empowered Internet of things (AIoTs) that converge sensing, communication, and computing to enhance wireless ubiquitous connectivity, data acquisition, and analysis capabilities. With full connectivity, the issue of data security in the new cloud-edge-terminal network hierarchy of AIoTs comes to the fore, for which blockchain technology is considered as a potential solution. Nevertheless, existing schemes cannot be applied to the resource-constrained and heterogeneous IoTs. In this paper, we consider the blockchain design for the AIoTs and propose a novel classified ledger framework based on lightweight blockchain (CLF-LB) that separates and stores data rights at the source and enables a thorough data flow protection in the open and heterogeneous network environment of AIoT. In particular, CLF-LB divides the network into five functional layers for optimal adaptation to AIoTs applications, wherein an intelligent collaboration mechanism is also proposed to enhance the across-layer operation. Unlike traditional full-function blockchain models, our framework includes novel technical modules, such as block regenesis, iterative reinforcement of proof-of-work, and efficient chain uploading via the system-on-chip system, which are carefully designed to fit the cloud-edge-terminal hierarchy in AIoTs networks. Comprehensive experimental results are provided to validate the advantages of the proposed CLF-LB, showing its potentials to address the secrecy issues of data storage and sharing in AIoTs networks.

1. Introduction

Current Artificial Intelligence Internet of Things (AIoTs) systems, 5G communications, and cloud computing are deeply integrated, which effectively enhances wireless ubiquitous connectivity, data acquisition, and analysis capabilities. In particular, in the IoT environment, various wireless networks, such as 5G, WiFi, and BLE, can achieve high-speed point-to-point communication and then conduct various functional subnetworks, enabling collaboration among heterogeneous nodes to support various types of services [1–3].

Through ubiquitous sensing and connectivity, all kinds of data can be efficiently collected and transported at high speed. Meanwhile, data security risks are also increasing. Specifically, in the new "cloud-edge-terminal" environment, although the central cloud has massive data storage capacity [4–7], it is unable to protect the terminal and user-side source data in an open environment and lacks a mechanism to promote peer-to-peer secure data sharing among users. During the process of data transportation, the data are vulnerable to malicious modification and duplication, and data rights are difficult to be guaranteed. To this end, there is an urgent need to study the mechanisms and methods for validating and managing the data rights throughout the life cycle in the AIoTs.

Blockchain technology conducts a network in which everyone can autonomously store transaction data anonymously and securely, is open, self-organizing, and difficult to be attacked. Furthermore, it can solve the abovementioned management issues of data rights, thus receiving increasing attention from industry and academia. In particular,

blockchain technology uses different policies that are replicated to each node as much as possible [8], open distributed storage network services [9], value incentive and flow capabilities [10], and self-organizing fault tolerance [11], in order to enable data protection mechanisms for open environments. However, traditional blockchain technology using a decentralized homogeneous model leads to a high volume of communication overheads. Moreover, disordered competition leads to high energy consumption of consensus. Thus, the traditional blockchain cannot take full advantage of the characteristics of the IoT to slice functional modules and flexibly deploy them to various layers so as to adapt to environmental constraints. Meanwhile, it is unable to classify and store different structural ledgers and inefficient book keeping and retrieval. Therefore, there is a need to design lightweight frameworks for flexible deployment in MEC.

A number of lightweight IoT-oriented blockchain designs have emerged recently. The new designs are able to basically achieve on-demand elastic scheduling of computing resources [12], weighing data transmission and storage between terminal networks, edge networks, and cloud platform networks so as to form blockchain storage capabilities that are suitable for this edge-computing architecture. Among them, simplification of the blockchain's consensus, ledger structure, P2P, and other mechanisms are the focus of the current research attention. Specifically, Liu et al. [13] study lightweight blockchain systems for the Industrial Internet of Things (IIoTs), propose green Co-PoW green collaborative consensus, and design LightBlock's lightweight data structure to simplify broadcast content which uses irrelevant block offload filters to reduce ledger blocks. However, the lack of systematic analysis of the cloud-edge-terminal collaboration limits scalability. Then, Liu [14] also proposes the heterogeneous and resource-limiting features of IoTs, constructs an independent Tornado P2P network layer, utilizes the current sensor network and management network hierarchical structure of IoT, and divides two levels to compute and pack macro-/microblocks, respectively, which uses space-structured ledger to extend the performance, reaching the maximum throughput of 3464.76 transactions per second. However, it is still energy-consuming of elections, synchronized consensus and ledger across the network, and full data storage ledger. Ferrag et al. initially investigate and propose future research areas such as dynamic adaptive security framework, social network and trust management, and specific blockchain infrastructure in terms of security goals, performance, limitations, computational complexity, and communication overheads [15]. However, there is a lack of viable framework recommendations, failing to propose specific functions of blockchain infrastructure and how to achieve orderly management in conjunction with edge computing.

In summary, the relevant research has achieved some innovations, which are limited by the current state of IoTs infrastructure. In addition, the new challenges we consider cannot yet be adequately addressed, such as the problem of efficient data storage and the coherent protection of privacy and data rights in AIoTs scenarios. In this paper, we propose a multiledger framework (CLF-LB) for future 5G/6G fully connected scenarios to address the problems of the current blockchain framework, such as no classification structure, no management, and disordered competition. Specifically, we propose a "cloud-edge-terminal" division of multilayer and multiclassification storage networks. Then, we lighten the blockchain ledger for local real-time processing. We also realize that orderly management can reduce workload by gradually reinforcing the security level on the demand. The results can take full advantage of the new architecture's full connectivity, intelligent perception, data portrayal, and other characteristics.

This research focuses on analyzing the real-time classification storage requirements of localized blockchains in the IoT network. Different ledger structures are constructed based on the stored data subjects and data rights.

Compared with the existing unimodal [10], unordered competitive consensus [16], and all-data storage blockchain for IoTs [14], CLF-LB has the following novel contributions and main advantages:

1.1. Multilayer and Multiclassification Framework. A hierarchical and clustered multiledger blockchain network, with a convergent approach that takes both local efficiency and the massive processing power of the cloud platform into account, focuses on the extraction and protection of data interests.

1.2. Real-Time Data Rights Ledger. A ledger daily inherits and implements regenesis strategies with forkless, fixed, prefabricated minute blocks.

1.3. Iterative Reinforcement of Consensus. A lightweight and incremental consensus can maximize the security in resource-constrained scenarios.

The remainder of this paper is organized as follows: In Section 2, we present related work and our contribution. In Sections 3 and 4, we propose our algorithms and models. The experimental results and a scheme comparison are described in Section 5. We make a conclusion in Section 6.

2. Related Work

In terms of decentralized blockchain design, unimodal networks lack authoritative mechanisms, and the competitive consensus strategy is adopted to ensure the consistency and correctness of each transaction on all ledger nodes; however, the resource consumption is quite high. The commonly used consensus algorithms include proof-of-work (PoW), proof-of-stake (PoS) [16], and delegated proof-of-stake (DPoS) [17]. The PoW mechanism consumes a large amount of energy, which severely limits the transaction throughput. In addition, PoS shortens the time it takes to reach consensus among nodes and avoids the massive waste of resources caused by mining; however, it undermines fairness by discouraging "poorer" participants and allowing the "richest" stakeholder to have complete control over the block generation.

Moreover, based on PoS, the DPoS mechanism solves the high-energy consumption problem of POW and avoids the "trust-balance" bias possible under PoS, but is not decentralized enough. In addition, the zero-knowledge proof algorithm [18, 19] simplifies the PoW and also increases the difficulty of attacks, which further enhances the privacy and security of consensus strings.

In terms of lightweight blockchain design, with the increasing number of IoTs devices, there are huge security risks associated with storing massive amounts of information. Due to high-energy consumption and large processing overhead, the existing blockchain architectures are not suitable for IoTs scenarios. A lightweight blockchain architecture can be achieved on the premise of ensuring data security and privacy through strategies such as heterogeneous environment adaptation, RAFT consensus [20], eliminating tailoring means such as miner and value records [21], and dividing subnets [14].

In terms of adaptation to heterogeneous environments, MEC has recently gained widespread attention for data processing, data analysis, and data storage in heterogeneous Internet of Things (H-IoTs) scenarios [22]. Furthermore, MEC takes full advantage of the computing power of edge nodes, greatly reducing the computational pressure on data centers and facilitating the storage and processing of big data. Due to the lack of management of distributed nodes, edge nodes are easy targets of hacking. Differential privacy-based machine learning strategies address the privacy issues in edge computing from both data aggregation as well as data mining [23]. Wang et al. [24] studied the integrated framework of computational offloading and interference management in MEC-enabled wireless cellular networks to support the accomplishment of indivisible computational tasks. Dinh et al. [25] show that it is possible to perform computational tasks without requiring prior system knowledge, but instead, with the help of emerging reinforcement learning (RL) algorithms that can optimally learn the dynamic computational triage strategy. Yan et al. [26] model the process of offloading and caching to ensure that both edge nodes and edge computing service providers obtain the maximum profit based on game theory and auction theory. Gong et al. [27] present an intelligent cooperative edge (ICE) computing in IoTs networks to achieve a complementary integration of AI and edge computing. Wei et al. [28] used a deep reinforcement learning algorithm, which combines RL method Q-learning with the deep neural network (DNN) to approximate the value functions for complicated control applications, and the optimal policy will be obtained when the value function reaches convergence. They find that the computation offloading and content caching achieve a better solution using localized AI.

In terms of tailoring adaptation, the transaction throughput of traditional blockchains is too low to meet the high scalability requirements of IoTs. The tailor-made lightweight blockchains simplify the processing of key modules and enhance the scalability of the blockchain without changing the model structure. Karlsson [29] proposed a blockchain with a directed acyclic graph (DAG) structure, which supports asynchronous concurrent writing of single-user transactions, thereby reducing the storage requirements of data, while enabling the tracking of data sources and the creation of shared tamper-proof data repositories. Since it is impossible to simultaneously ensure consistency, availability, and partition tolerance in a blockchain system, the system's design often needs to weaken the guarantee of a particular feature. Karlsson also adopted a partition-tolerant blockchain "Vegvisir blockchain" for power-constrained, network-connected IoTs environments. Hassija et al. [30] proposed an IoTs network based on a lightweight blockchain protocol using the Tangle data structure, which replaces the traditional chained data structure to record transactions in the network in a secure and scalable manner. The model is highly scalable as it does not require extensive computation to add transactions to a block, nor does it require any transaction compensation charges. As the data in a block grows dramatically, it can incur significant communication and storage overhead. Therefore, a fragmented ledger is used to store the relevant detailed data, and the storage structure of the shared ledger is lightened to achieve reliable and traceable event analysis with minimal storage and processing overhead [31]. Based on this, Yang W [32] proposed a social-based data simplification approach, a directed acyclic graph (DAG) lightweight blockchain model, where each node stores only the data of interest and ignores irrelevant data to reduce the number of duplicate data in the blockchain. The experimental results showed that the model saves 97.13% of the storage space. With a lightweight blockchain, Xin Jiang et al. [33] proposed a new blockchain-based authentication protocol for WLAN mesh security access. It takes the user's authentication request as a transaction, considers all the authentication records in the mesh network as the public ledger and realizes the effective monitoring of the malicious attack. For ensure the security of data transmission in IoTs, Hui et al. [34] applies a new chaotic secure communication scheme to address the security problem of data transmission. The scheme is based on the synchronization of different-structure fractional-order chaotic systems with different orders.

Our scheme proposes using heterogeneous communication chips to directly extract metadata and equity data for uplink, balancing security, and resource consumption to achieve a new hierarchically deployed multimodal blockchain network.

3. System Model

In this section, we take the AIoT scenario as a case study of the CLF-LB scheme. Next, we give a detailed description of the essential multimodal network. Finally, we design a light block structure called regenesis. Table 1 lists the explanation of the symbols associated with the CLF-LB scheme.

It is known that Bitcoin and Ethereum are already severely limited by the amount of data because there are too many historical blocks. Hundreds of gigabytes of data and days-long block download tasks have prevented new full-featured nodes from being added. The workloads for data traceability, package validation, and energy consumption are

Table 1: Notations in the CLF-LB scheme.

Symbol	Notation
CRS	Common reference strings set
HCF	Hierarchical and classified framework
MNET	Multimodal network
AN	Access control network
CN	Wireless cache network
DN	Decision network
PN	Proof-of-work network
SN	Storage network
BRG	Block regenesis
TBLOCK	Temporary timestamp blocks
RBLOCK	Reinforcement emphasis blocks
CBLOCK	Permanent chain blocks
RPOW	Iterative reinforcement of proof-of-work
EDCC	Extract data rights on the chain by chip
BT	Bittorrent
BLE	Bluetooth low energy
WMN	Wireless mesh network
Zk-SNARKs	Zero-knowledge succinct noninteractive argument of knowledge algorithms

increasing. Light weighting the data that need to be processed in real time on the same day is a key point for the adaptability of blockchain technology to IoTs environments.

3.1. Hierarchical and Classified Framework. The proposed framework is depicted in Figure 1. In IoTs sensors, multimode communication chips can be used as a specific infrastructure to perform uplink processing directly. The smart devices are equipped with intelligent grouping and application access control; they also participate in blockchain services such as caching, packaging, and PoW task processing, as well as providing multiledger cluster caching services.

We implement a local-area blockchain network at the network-edge layer, lightly pregenerate a total of 1440 fixed blocks per minute, and regenerate previous blocks daily. We elected high-performance nodes to run multiple ledgers, schedule the PoW network to intelligently decompose tasks, and assign them to idle nodes for execution.

In a cloud storage system, raw data and metadata such as data rights and historical blocks are stored, and the daily chain is compressed into a single block that is concatenated day-by-day based on the structure of the public chain and PoW consensus.

To achieve a timely and flexible security level, we define the TBLOCK process of PoW for small-scale AIoT to find nonce as follows:

$$\begin{aligned}&\text{SHA256}\,(\text{SHA256}\,(\text{gensis meta data} + \text{block header}\\&\quad + \text{nonce})) < \text{target}\,(20\,\text{bit zero}).\end{aligned} \tag{1}$$

The indicators of difficulty and resource utilization control the effectiveness of the PoW network. For the process of confirming block to writing into the blockchain, we sample every ten minutes to strengthen the RBLOCK, defining the PoW as follows:

$$\begin{aligned}&\text{SHA256}\,(\text{SHA256}\,(\text{block header} + \text{root of merkel tree}\\&\quad + \text{nonce})) < \text{target}\,(24\,\text{bit zero}).\end{aligned} \tag{2}$$

Eventually, a CBLOCK storage in a cloud system should merge 1441 fixed blocks. It builds a new Merkel tree with the hash of each block. We define the PoW as follows:

$$\begin{aligned}&\text{SHA256}\,(\text{SHA256}\,(\text{block header}\\&\quad + \text{root of merkel tree} + \text{nonce}\,)) < \text{target}\,(28\,\text{bit zero}).\end{aligned} \tag{3}$$

3.2. Multimodal Network. We build a hierarchical and clustered multiaccount blockchain network for the three-layer structure, which is composed of an IoTs terminal, edge network, and cloud platform. We delineate five categories of different characteristics and multimodal regions to select the appropriate data processing network so as to match the appropriate participating nodes' networking in different environments.

We distributed the deployment of five different kinds of node networks in three layers to collaborate on data protection services as follows:

3.2.1. A Network (Access Control Network). This network is used mainly to strengthen authentication services in IoTs and protect data [35, 36]. It utilizes blockchain's end-to-end zero-knowledge proof for verification and consensus, to defend against network layer attacks and malicious tampering.

3.2.2. C Network (Wireless Cache Network). The C network is a real-time transaction caching network that clusters transactions in terms of ledger type, caches real-time transactions, and realizes batch verification of transactions.

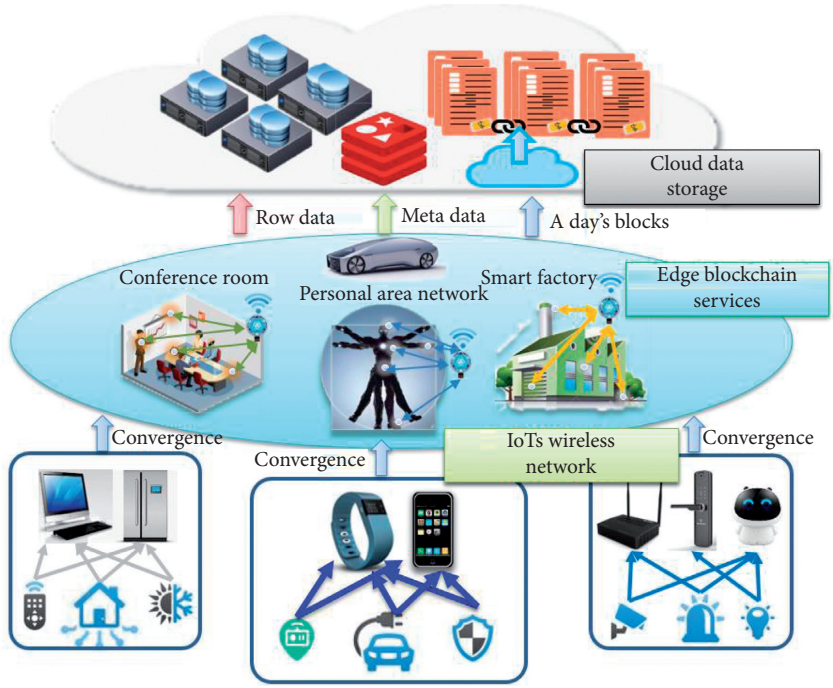


Figure 1: Hierarchical classified blockchain storage based on edge computing.

3.2.3. D Network (Decision Network). This is a local, lightweight, and packaged block network. We pack multichain block data in parallel across multiple centers and store equity data. This network realizes lightweight transmission, computation, and storage of data based on the requirements of the IoTs resource-constrained environment. Once the data are packaged, the cache will be cleaned periodically, and the real-time data in the backup cycle, and daily block formation files will be asynchronously uploaded to cloud storage.

3.2.4. P Network (PoW Network). This is a task-oriented PoW computing network. The network scheduling approach combines computing resources for producing proof-of-workload strings and discovering trusted and efficient nodes. A self-organizing approach is implemented to manage the nodes and addresses of the distributed network and realize the optimization of hierarchical connections.

3.2.5. S Network (Storage Network). The S network is a massively distributed storage network on the cloud platform designed to achieve massive and efficient storage of backup data, thereby solving the capacity constraint problem. The network stores the daily packed backup block files and data files. The blocks are concatenated into a permanent super chain via block references for multiple cycles. The PoW consensus and block-based Merkel Tree are used to protect integrity.

Our model is constructed from the tuple (A, C, D, P, and S). C->D->S cyclic iterations are used to accumulate historical operational data for subsequent decision analysis. The C and P networks collaborate to provide the basis for judging pending tasks and processing capacity. In the D network, intelligent nodes can dynamically assign tasks based on task queue length and adjust the difficulty of workload calculations. The D and S networks collaborate to balance network occupancy and local storage volume thresholds and are responsible for asynchronous gradual offloading of data to cloud storage. The P and D networks are mutually constrained to optimize the control of the D network by participating in the voting election process of the nodes.

In particular, the collaborative process of C, D, and S requires constant interaction with the environment to obtain multiparty states. It is a complete-information and cooperative-game model of the supply-chain pipeline. We define the following reward function (4) that can train the Deep Reinforcement Learning (DRL) model optimizing actions policy balancing the utilization of storage and network resources. The expected resource-occupancy rate is stable in the range of 30% to 70%. The corresponding positive reward is the proportional value of the positive rate of the change generated by the scheduling algorithm, and vice versa. When the occupancy rate is more than 70%, the model should accelerate the offloading action. When the occupancy rate is less than 30%, the model should increase the new tasks. The defined offload is the reward generated by the offloading action; the defined upload is the reward generated by the adding task action; and the defined loss is the state value to be deducted when the computing, storage, and network resources exceed 70% of the threshold.

$$R = \text{offload}(\text{ao}) + \text{upload}(\text{at}) - \text{loss}(\text{over\_cpu}, \text{over\_storage}, \text{over\_net}). \quad (4)$$

The matching DRL method, asynchronous advantage actor-critic (A3C) [37], can iteratively optimize the

scheduling strategy for hierarchical caching and transmission of data. The advantage function A (s, a) trains the actor-police model to maximize the benefits of the action. The loss function Q (s, a)-V (s) trains the critic-value model to evaluate the value of the current state. Finally, a unified model can be trained to output the probability distribution of actions and the value of the state, respectively. Using the loss function (5) composed of the following three parts (6–8) such as policy loss, value loss, and regularization with policy entropy, the deep learning framework minimizes this summarized loss.

$$L = L_\pi + \alpha L_v + \beta L_r, \tag{5}$$

$$L_\pi = -\frac{1}{n}\sum_{i=1}^{n} A(s_i, a_i) \cdot \log \pi(a_i|s_i), \tag{6}$$

$$L_v = \frac{1}{n}\sum_{i=1}^{n}\left(\left(\sum_{j=0}^{k-1}\gamma^j r_{i+j} + \gamma^k V(s_{i+k})\right) - V(s_i)\right)^2, \tag{7}$$

$$L_r = -\frac{1}{n}\sum_{i=1}^{n} H(\pi(s_i)). \tag{8}$$

Note that the collaboration between D and P is a typical two-stage Stackelberg game. The model needs to maximize utility for flexibly control workload while adapting the appropriate level of security. On the supply side, P maximizes the reception and completion of tasks and accumulates the workload for improving the level gradually. On the demand side, D needs to maximize scheduling reward and accumulates voting credit by honestly recording the contributions of the worker in the P network.

3.3. Block Regenesis. The CLF-LB scheme includes a mechanism to regenerate blocks on a daily basis by collaboration. The six-step processing flow of the data stream is shown in Figure 2.

3.3.1. Phase of Aggregation of Transaction Records

Step 1. The security network manages access control and privileges of its applications. Here, we organize IoT nodes, manage identities and permissions, and provide access control mechanisms. Mainly, we strengthen authentication services in the IoT to protect data and set up public reference strings to reinforce confidentiality. Next, we schedule the PoW network according to the security-level requirements to produce the PoW required for the corresponding level of authentication string.

Step 2. The real-time transaction caching network receives the broadcasted transaction records. Various types of nodes in the IoT can simultaneously send the certificate of deposit transactions to all nodes in the caching network via a wireless broadcast mechanism so as to cache real-time transactions. A transparent infrastructure layer is formed through chip-cured communication management, data entitlement extraction, and an uplink process. To be able to respond to end users in real time, an in-memory caching network is built on the smart-device or edge-gateway side. Real-time clustering is used to cache different transaction records and real-time streaming data to verify data consistency, integrity, and authenticity.

3.3.2. Phase of Blockchain-Service Processing

Step 3. The local lightweight block network sorts blocks. Multiple local chains storing different ledgers will elect the appropriate central node by RAFT consensus. The multiple centers pack multiple chains and different classifications of block data in parallel and store equity data. The central nodes first lightly prefabricate fixed blocks per minute of the current day and then regenerate the blocks on a daily basis after inheriting the previous day's balance pencils and other summary information.

We assign tasks to the PoW network to produce workload-proof strings, ranging from easy to difficult. The transaction records in the cache are sorted by timestamp order and recorded into blocks with different sequence numbers at regular intervals every minute. The status is determined 10 minutes later; the final block to be packaged is fixed and generated for posting to the following nodes.

Step 4. The local lightweight block network parcels blocks. Subsequent nodes in the RAFT network accept the blocks to be packaged for local storage. This step includes the P2P network method, which actively propagates the block header chain table to all participating PoW nodes, and the master node modifies the status to pack complete according to the number of acknowledgments in the RAFT network. This step also provides download services for individual blocks and individual transaction records.

3.3.3. Phase of Storing in Cloud for Recycling

Step 5. The periodic tasks submit data to the cloud storage network. The video stream in the IoTs needs to be segmented and cached locally. The corresponding metadata and data entitlements are recorded in the block, and after completing the packaging and archiving, the task is submitted asynchronously. The backend thread progressively advances the task and is responsible for uploading each segmented data to the cloud storage network. After uploading, the local backup file can be deleted.

Step 6. The leader packs the chain file daily submitting it to the cloud storage network. Historical data in daily blocks are packaged and compressed into a chain file, as a block on the permanent super chain, which is then uploaded to the cloud storage network. The daily packaged backup block file is stored in the cloud and concatenated into a permanent super chain by block referencing for multiple cycles, thereby protecting the integrity and preventing data tampering through PoW consensus. As the public chain policy,

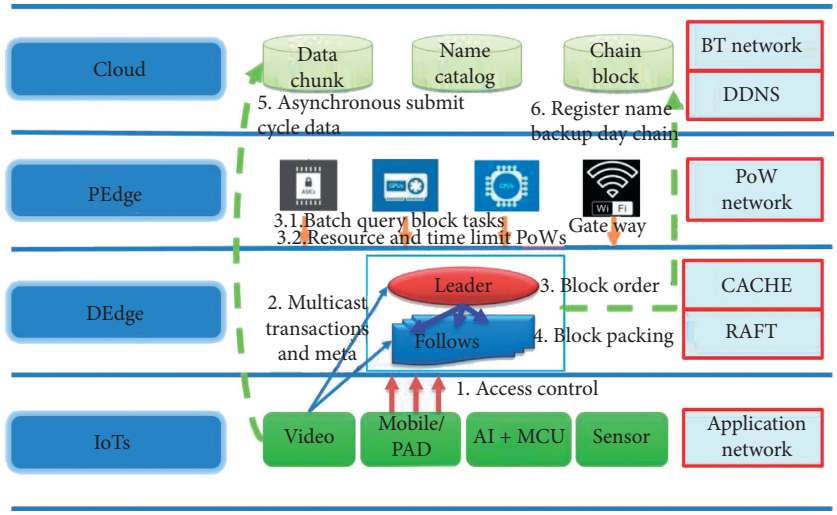


Figure 2: Process of data block regenesis.

consensus concatenates multiday blocks to ensure that data for more than six days is essentially impossible to roll back. Super chains are stored in the cloud distributed storage, reducing local storage space pressure and redundancy.

4. Dynamic Adaptivity

4.1. Iterative Reinforcement of PoW. The primary step is to build a dynamically joinable and task-oriented PoW computational network. Participating nodes are able to perform tasks according to their capabilities, either passively or proactively, by querying the blocks to be packaged and submitting workload proofs on a regular basis. The network provides computing power services for prepacking local blocks, applying secure access tokens, PoW consensus, and other secure processing based on PoW.

Unlike PoW networks that maximize network workload goals and set the corresponding difficulty, RPoW determines the benchmark workload by network capacity and security-level definitions and without using PoW to compete for book-keeping rights and rewards. The lightweight and incremental batch production of workload-proof strings gradually increased the verification difficulty to maximum security difficulty in resource-constrained scenarios while meeting the constraints of the task's deadline.

As a security support mechanism, computational nodes focus on participating and providing capacity indicators. Node trustworthiness is enhanced through multidimensional indicators such as capacity, participation time, and contribution. Open mechanisms for sharing computing power and contributing resources can improve nodes and address the management of distributed networks. With the participation of more trustworthy nodes, the connection management capability of each network can be improved, and the synchronization data capability of the network can be improved so that the packaged nodes can be compared and selected.

The reinforcement consensus makes the P network more flexible for the security goal by asynchronous iterating. The D network makes the decision for high-effect utilize the resources of the P network through intelligent perception. The optimization goal of the scheduling algorithm is to maximize the computational difficulty (CD) of the consensus string. The constraints include limited time (LT), limited resources (LR), and the number of tasks that will be submitted at the next time frame (TS). The main parameters with which the schedular judges the current difficulty are the priority sequencing queue length of the task (QT) and the current basic difficulty factor (BF). By accumulating the historical multicycle time (CT) and the number of transactions (CN), the peak time-frame range (PT) by daily predicting is taken as the fixed BF to maintain the minimum input task. Using the current observation value, the Kalman filter algorithm can continuously smooth the estimated value and predict the transaction number (PN) in each minute, which is the basis for scheduling to increase the workload. The optimal workload of the ith minute is calculated by the following formula:

$$\begin{aligned} \max \quad & CD[i] = A(BF, QT) + \gamma(S(P[i])) + \gamma(S(P[i+1] \\ & \quad + \cdots \gamma S(P[n]))), \\ & LT = \text{True}, \\ \text{s.t.} \quad & LR = \text{True}, \\ & TS = \text{True}. \end{aligned} \tag{9}$$

The framework of the algorithm is shown in Figure 3.

4.2. Extract Data Rights on the Chain by Chip. Based on the blockchain computation and connection embedded in the chip, we realized the automatic uplink processing. Embedded in the 5G/WiFi/BLE module chip of IoTs, processing such as trusted registration and statistical service data flow, uplink communication processing, and security cryptographic signature is used to achieve anticounterfeit control and an efficient real-time uplink mechanism. The data generated by resource-constrained terminals can be directly

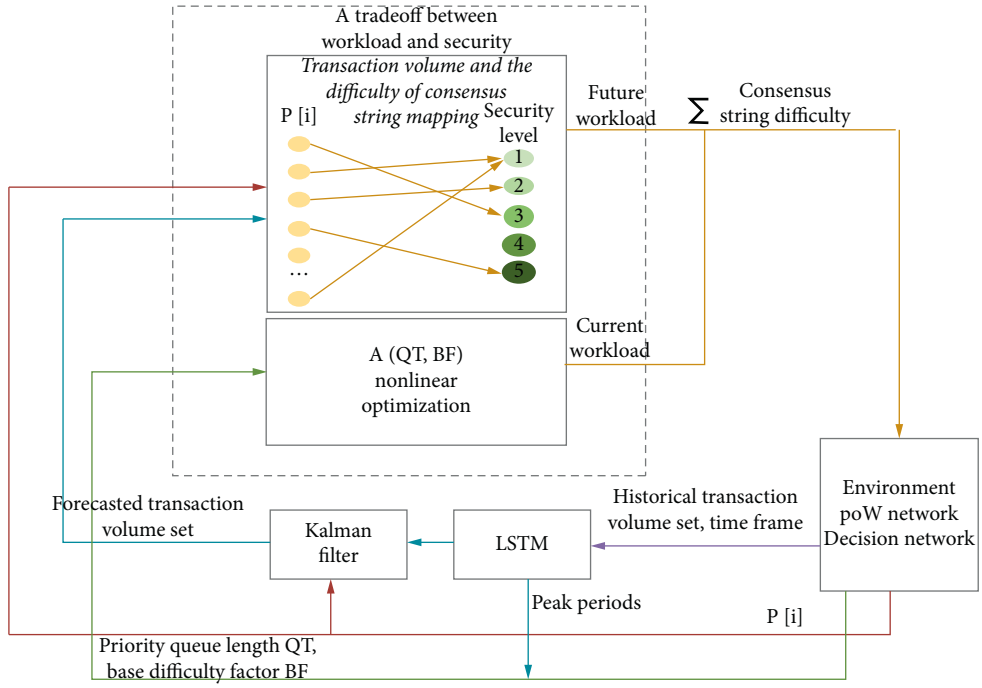


Figure 3: The framework of dynamic adjustment of difficulty factors.

uploaded through the blockchain module embedded in the chip, without the need for application development and occupying terminal resources.

The first step to realizing this is trusted registration processing, registering the identity within the application security network. The unique identification and location information of the chip can be used as the registration evidence of the service to construct the identity of each service for which the communication chip assumes data transmission, including address, public key, and private key.

The second step is to authorize the establishment of an access relationship within the application security network, apply an authentication token, and access the data service.

The chip can classify service data traffic information according to the predefined service types and regularly count the transmitted service data traffic information, including the service object, service start time, service traffic statistics, and service duration. Using the identity of the service, the subsequent uplink processing is triggered at regular intervals. The data volume and service statistics transmitted by the chip are deposited daily into the proof blockchain. The contextual information of these services is trustworthy and difficult for other devices to falsify.

The last step is that the terminal application extracts user rights and privacy information at the data collection stage, collects statistics regarding the characteristics of the collected data, forms a transaction record, calls the chip API interface of the uplink process, and triggers the uplink processing operation of the chip.

The specific uplink communication process is as follows:

(1) The system utilizes a neighbor discovery protocol for wireless communication and discovers and manages various types of cached service addresses in the vicinity. Then, it evaluates and ranks them in terms of service quality.

(2) The user information and privacy fields are encrypted, and the uplink data are signed.

(3) The RPC protocol format is used for encapsulating uplink transactions. The system picks the first n cache service nodes based on the COAP protocol for IoT. It then uses the packet broadcasting protocol to broadcast and send transaction records to multiple cache server addresses.

5. Evaluation

5.1. Implementation. We build a multimachine Fabric1.4 network environment based on RAFT consensus, using the REST API to store evidences, trace the source by caches, and trace by blocks. These services can also trace data replication, along with the addition and deletion of data rights.

5.2. Testbed. The devices comprising the network structure and the computing power of each node are shown in Table 2.

5.3. Influence of the Number of Nodes. To evaluate the influence of the number of nodes, we execute the evidence storage service 1,000 times in each scenario. As shown in Figure 4, the number of participating nodes gradually increases from one node to 40 nodes, which increases the average latency of evidence storage and sharply decreases the performance of TPS.

Table 2: Experimental devices.

Type	Specification	Hashrate (MH/s)	Concurrency	Number
MCU	ARM cortex-A7 CPU @ 1.60 GHz 4C/1G	0.16	2	1
Edge devices	Intel (R) xeon (R) platinum 8269CY CPU @ 2.50 GHz 2C/8G	1.1	1	1
	Intel (R) xeon (R) gold 6278C CPU @ 2.60 GHz 4 C/8G	1.2	3	1
	Intel (R) core (TM) i7-6700HQ CPU @ 2.60 GHz 4 C/8G	1.3	4	1
Server	Intel (R) xeon (R) E5 CPU @ 3.20 GHz 16C/32G	1.4	16	1

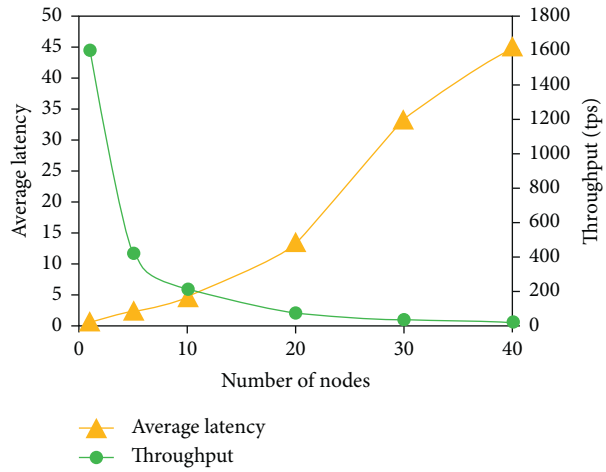


Figure 4: Performance of evidence storage with varying number of nodes.

Therefore, to avoid performance degradation, we adopt the strategy of dividing multiple ledgers to reduce the number of packaging nodes in each subnet.

We design a strategy for voting a master node daily. Having a master node means that we can avoid disordered competition, and this is the only way to maximize serial performance. Hence, the framework implements parallel block data for each master node through hierarchy and classification.

5.4. Performance of the Cache Network. We use the cache method of key-value storage to record the query time of traceability data under different cache sizes. The results are shown in Table 3. As the cache size increases gradually from 10 to 100,000 groups, the time taken to trace is not affected, and the time consumption is always short because we use the unique key identifier. The average trace time was 26.298 ms, and the standard deviation was within [3.01, 3.46] ms.

According to the abovementioned analyses and test results, we select 5000 groups of data to test the C network of this paper and count the probability distribution of the query trace time, as shown in Figure 5. The probability curve obeys the normal distribution. The mathematical expectation of the query trace-time μ is 26.298 ms, and the standard deviation δ is 3.27 ms. The values within the standard deviation range account for 86.12% of the total values. Therefore, the C network can easily create and manage the data in the cache, thus providing an efficient batch verification mechanism for real-time transactions, which is a key component of lightweight blockchains.

Table 3: Query time performance of the cache network.

Cache sizes	Average (ms)	Max (ms)	Min (ms)	Std
10	26.50	46	22	3.46
100	26.30	43	21	3.44
1,000	26.31	45	21	3.13
10,000	26.34	44	22	3.30
100,000	26.04	43	21	3.01

5.5. Inspections of Daily Regenesis. For scenarios of 100, 500, 1000, and 2000 transactions, we compare the impact of having a different number of blocks on traceability query processing. The results are shown in Table 4. The original scheme I controls the generation of 10 blocks per 100 transactions, whereas the improved scheme II generates two blocks for every 100 transactions.

By reducing the number of local blocks, scheme II makes most of the data traceable in the same block, reducing the trace latency of cross block. The daily regeneration strategy can control the number of local blocks and improve the query efficiency of the day. Especially, when new nodes join the network, the scheme improves the processing of block packing, broadcasting, and transaction and reduces the network time consumption and traffic. The experimental results are shown in Figure 6.

Table 4: Traceability performance in relation to the number of blocks.

Trans	100	500	1,000	2,000
Original I blocks	10	10×5	10×10	10×20
Original I times (ms)	557	2,717	5,338	10,687
Improved II blocks	2	2×5	2×10	2×20
Improved II times (ms)	510	2,503	4,953	9,929
Improved II promoted	8.44%	7.88%	7.21%	7.09%

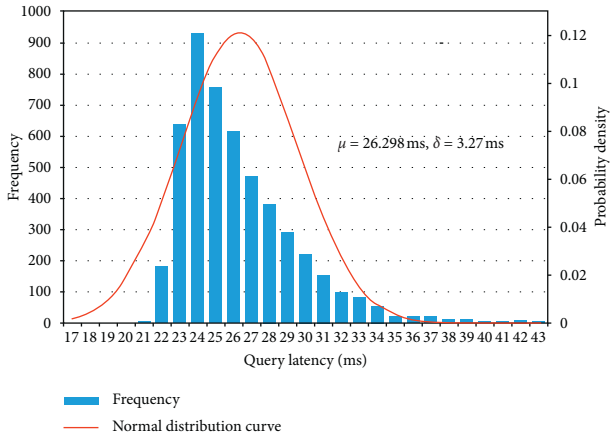


Figure 5: The probability distribution of traceability query time in the C network.

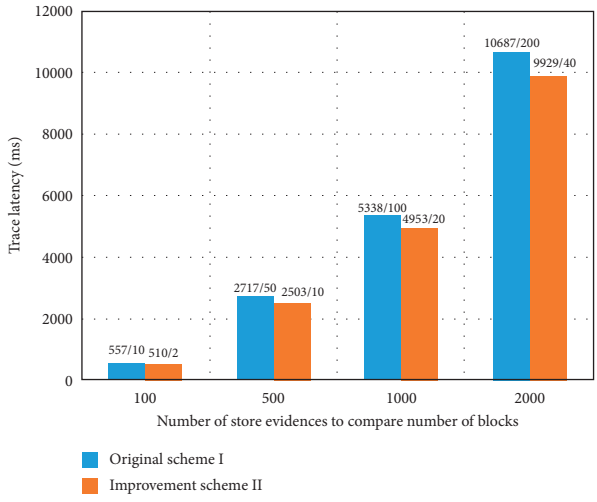


Figure 6: Traceability performance with different number of blocks.

6. Conclusions

In this paper, we propose a lightweight blockchain that can be deployed in heterogeneous, multisource, and multimodal IoTs environments. By dividing the network into different models, intelligent collaboration is promoted. Thereby, the framework can support the daily regeneration of local blocks to lightens the local storage data. Specifically, the fixed structure improves real-time processing ability. To keep a tradeoff between security and cost, the RPoW algorithm iteratively increases computational complexity on demand. Meanwhile, within the communication chips, the data directly upload to the blockchain that sorts and stores the multisource data in the classified ledgers. Finally, the experiments demonstrate that the framework reduces resource consumption, enabling blockchain service in MEC for throughout data flow protection.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Acknowledgments

This work was supported by the Key Technologies of Trusted Sharing of Multi-source and Multi-modal Data Based on Blockchain Project under Grant No. 2019YFB2102403.

References

[1] M. I. Poulakis, A. G. Gotsis, and A. Alexiou, "Multicell device-to-device communication: a spectrum-sharing and densification study," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 85–96, 2018.

[2] C. E. Casetti, C. F. Chiasserini, Y. Duan, P. Giaccone, and A. Perez Manriquez, "Data connectivity and smart group formation in wi-fi direct multi-group networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 245–259, 2018.

[3] Bluetooth, *Bluetooth Core Specification, 5.2*, Bluetooth Special Interest Group Std, Kirland, WA, USA, 2020.

[4] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. Long, and C. Maltzahn, "Ceph: a scalable, high-performance distributed file system," in *Proceedings of the 7th Conference on Operating Systems Design and Implementation (OSDI '06)*, pp. 307–320, Seattle, WA, USA, November 2006.

[5] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 29–43, 2003.

[6] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies*, pp. 1–10, Incline Village, NV, USA, May 2010.

[7] Y. Chen, C. Li, M. Lv, X. Shao, Y. Li, and Y. Xu, "Explicit data correlations-directed metadata prefetching method in distributed file systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2692–2705, 2019.

[8] R. Van Renesse, D. Dan, V. Gough, and C. Thomas, "Efficient reconciliation and flow control for anti-entropy protocols," in *Proceedings of the 2nd Workshop on Large-Scale Distributed Systems and Middleware (LADIS '08)*, Association for Computing Machinery, New York, NY, USA, September 2008.

[9] J. Benet, "IPFS-content addressed, versioned, p2p file system," 2014, https://arxiv.org/abs/1407.3561.

[10] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, HN Publishing, Guldborg, Denmark, 2008, https://bitcoin.org/bitcoin.pdf%20.

[11] P. Maymounkov and D. M. Eres, "Kademlia: a peer-to-peer information system based on the xor metric," in *Proceedings of the 2002 Revised Papers from the First International Workshop on Peer-To-Peer Systems*, Springer-Verlag, Cambridge, MA, USA, March 2002.

[12] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT security: ongoing challenges and research opportunities," in *Proceedings of the 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, pp. 230–234, Matsue, Japan, November 2014.

[13] Y. Liu, K. Wang, Y. Lin, W. Xu, and " Lightchain, "$\mathsf {LightChain}$: a lightweight blockchain system for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3571–3581, 2019.

[14] Y. Liu, K. Wang, K. Qian, M. Du, and S. Guo, "Tornado: enabling blockchain in heterogeneous internet of things through a space-structured approach," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1273–1286, 2020.

[15] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the internet of things: research issues and challenges," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2188–2204, 2019.

[16] S. Fahad, "Blockchain without waste: proof-of-stake," *The Review of Financial Studies*, vol. hhaa075, 2020.

[17] L. Jiang, S. Xie, S. Maharjan, and Y. Zhang, "Joint transaction relaying and block verification optimization for blockchain empowered D2D communication," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 828–841, 2020.

[18] D. Gabay, K. Akkaya, and M. Cebe, "Privacy-preserving authentication scheme for connected electric vehicles using blockchain and zero knowledge proofs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5760–5772, 2020.

[19] D. Han, X. Du, and Y. Lu, "Trustworthiness and a zero leakage OTMP-P2L scheme based on NP problems for edge security access," *Sensors*, vol. 20, no. 8, p. 2231, 2020.

[20] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proceedings of the 2014 ATC USENIX*, Philadelphia, PA, USA, June 2014.

[21] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: the case study of a smart home," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops*, pp. 618–623, (PerCom Workshops), Kona, HI, USA, May 2017.

[22] Z. Yan, W. Yuan, M. Hassnaa, D. H. K. Tsang, L.-G. Albert, and U. Javaid, "Multi-access mobile edge computing for heterogeneous IoT," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 12-13, 2018.

[23] M. Du, K. Wang, Y. Chen, X. Wang, and Y. Sun, "Big data privacy preserving in multi-access edge computing for heterogeneous internet of things," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 62–67, 2018.

[24] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, 2017.

[25] T. Q. Dinh, Q. D. La, T. Q. Quek, and H. Shin, "Distributed learning for computation offloading in mobile edge computing," in *Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, July 2018.

[26] Y. Yan, Y. Dai, Z. Zhou, W. Jiang, and S. Guo, "Edge computing-based tasks offloading and block caching for mobile blockchain," *Computers, Materials & Continua*, vol. 62, no. 2, pp. 905–915, 2020.

[27] C. Gong, F. Lin, X. Gong, and Y. Lu, "Intelligent cooperative edge computing in internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9372–9382, 2020.

[28] Y. Wei, Z. Wang, D. Guo, and F. Richard Yu, "Deep q-learning based computation offloading strategy for mobile edge computing," *Computers, Materials & Continua*, vol. 59, no. 1, pp. 89–104, 2019.

[29] K. Karlsson, W. Jiang, S. Wicker et al., "Vegvisir: a partition-tolerant blockchain for the internet-of-things," in *Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1150–1158, IEEE, Vienna, Austria, July 2018.

[30] V. Hassija, V. Chamola, S. Garg, D. N. G. Krishna, G. Kaddoum, and D. N. K. Jayakody, "A blockchain-based framework for lightweight data sharing and energy trading in V2G network," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5799–5812, 2020.

[31] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, "Block4Forensic: an integrated lightweight blockchain framework for forensics applications of connected vehicles," *IEEE Communications Magazine*, vol. 56, no. 10, pp. 50–57, 2018.

[32] W. Yang, X. Dai, J. Xiao, and H. Jin, "LDV: a lightweight DAG-based blockchain for vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5749–5759, 2020.

[33] X. Jiang, M. Liu, C. Yang, Y. Liu, and R. Wang, "A blockchain-based authentication protocol for WLAN mesh security access," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 45–59, 2019.

[34] H. Hui, C. Zhou, S. Xu, and F. Lin, "A novel secure data transmission scheme in industrial internet of things," *China Communications*, vol. 17, no. 1, pp. 73–88, 2020.

[35] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[36] H. Chen, W. Wan, J. Xia et al., "Task-attribute-based access control scheme for iot via blockchain," *Computers, Materials & Continua*, vol. 65, no. 3, pp. 2441–2453, 2020.

[37] V. Mnih, A. P. Badia, M. Mirza et al., "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33rd International Conference on Machine Learning*, New York, NY, USA, June 2016.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6643034, 14 pages
https://doi.org/10.1155/2021/6643034

Research Article

Output Feedback NCS of DoS Attacks Triggered by Double-Ended Events

Xinzhi Feng,[1] **Yang Yang**,[1] **Xiaozhong Qi**,[1] **Chunming Xu**,[2] **and Ze Ji**[3]

[1]*School of Electronic Information Engineering, Changchun University of Science and Technology, Changchun 130022, China*
[2]*Changchun Shikai Science & Technology Industry Co. Ltd., Changchun 130012, China*
[3]*School of Engineering, Cardiff University, Cardiff, UK*

Correspondence should be addressed to Yang Yang; cloneyang@126.com

Received 27 December 2020; Revised 24 January 2021; Accepted 27 January 2021; Published 11 February 2021

Academic Editor: Liguo Zhang

In recent years, the research of the network control system under the event triggering mechanism subjected to network attacks has attracted foreign and domestic scholars' wide attention. Among all kinds of network attacks, denial-of-service (DoS) attack is considered the most likely to impact the performance of NCS significantly. The existing results on event triggering do not assess the occurrence of DoS attacks and controller changes, which will reduce the control performance of the addressed system. Aiming at the network control system attacked by DoS, this paper combines double-ended elastic event trigger control, DoS attack, and quantitative feedback control to study the stability of NCS with quantitative feedback of DoS attack triggered by a double-ended elastic event. Simulation examples show that this method can meet the requirements of control performance and counteract the known periodic DoS attacks, which save limited resources and improve the system's antijamming ability.

1. Introduction

Scholars have recently devoted much energy to studying network security and control stability of network systems against malicious attacks, including denial of service (DoS) attacks, replay attacks, and spoofing attacks. The research results on different types of network attacks can be obtained in the literature [1–5].

Among all kinds of network attacks, the DoS attack is considered the most likely type of network attack to impact the performance of NCS [6] significantly. In the past few years, the stability analysis of NCS under DoS attacks has been reported in the literature [7, 8]. Besides, because the event-triggered mechanism can be used to reduce the transmission of information without reducing the overall system performance, in recent years, more and more literature have used the event-triggered method to study the control stability of NCS under DoS attacks [9]. For example, in [6], the author, for the first time, investigated the stability analysis of event-triggered networked linear continuous-time systems under known pulse width modulation DoS attacks. Inspired by this work, the author proposes a more general DoS attack model in [10], in which the DoS attack signal is only constrained by DoS frequency and duration. Some extensions are also reported based on this idea, such as the dynamic output feedback controller [11] and distributed NCS [12].

An event trigger mechanism dependent on the Lyapunov function is proposed for nonlinear continuous systems in [13]. Eqtami et al. [14] use ISS's technology to propose a self-triggering mechanism and a new event triggering mechanism for nonlinear discrete systems. A new event mechanism is proposed for nonlinear systems through Lyapunov in [15]. The asymptotic stability of the system is verified by using the distributed event trigger mechanism in [16], and the small gain theorem for the ISS-Lyapunov function [11] solved the resource consumption and resilient control strategy design of NCS subjected to malicious DoS attacks. In the presence of DoS attacks, the proposed system framework can still guarantee a strictly positive and lower bound on the event time. A system design framework is proposed for the output-based dynamic event trigger control

(ETC) system under DoS attacks. De Persis and Tesi [17] studied the effective control strategy of nonlinear systems under a DoS attacks and provided the character of the maximum percentage of time that feedback information can be lost without causing feedback instability. It is based on the design of the event's elastic control strategy, and the fixed pattern is used as an exact representation of the interval activity interval. The controller triggered by the event has made a clear representation of the minimum cross-sampling time of the controller. Su et al. [18] proposed incremental algorithms for rapidly evolving the network strategies, and the redundancy rules are further processed.

Focusing on defensive strategies against network attacks in cyberphysical system (CPS), Tian et al. [19] proposed both low- and high-interaction honeypots into CPS as a security management tool deliberately designed to be probed, attacked, and compromised. To improve the attack detection capability of content centric network (CCN), Xu et al. [20] proposed a way of interest flooding attack (IFA) making use of the characteristics of self-similarity of traffic and the information entropy of content name of interest packet.

Distributed Denial-of-Service (DDos) attack has become one of the most destructive network attack. Cheng et al. [21] proposed a detection method of DDos' attacks based on generalized multiple kernel learning combining with the constructed parameter R, and the proposed method can effectively detect DDos' attacks in complex environments with a higher rate and lower error rate. Chen et al. [22] provided the active approach based on the integrated entropy calculations to detect DDos' attack. It is a lightweight approach could be applied in mobile device.

2. Preparation

2.1. LMI. The research on LMI (linear matrix inequalities) has been widespread. When it is used to solve control problems, LMI will be used to simplify the practical issues involved. Here are some definitions of LMI and many practical issues of LMI implementation used in later sections [23]. In general, the specific LMI, formula is

$$G(x) = G_0 + x_1 G_1 + \cdots + x_m G_m < 0, \tag{1}$$

where $x_1, x_2, \ldots, x_{m-1}, x_m$.. is the measure decision variable of LMI. $x = (x_1, x_2 \cdots x_{m-1}, x_m)^T \in R^m$ is the decision vector $G_i = G_i^T \in R^{n\times n}$, $i = 0, 1, \ldots, m$ is a symmetric matrix. Equation (1) is the general form of LMI, but through the exploration of scholars, most of the content of its solution is a variable matrix. If the following Lyapunov inequality

$$G(X) = XA + A^T X + Q < 0, \tag{2}$$

where $Q \in R^{n\times n}$ is a real symmetric matrix, $A \in R^{n\times n}$ is a constantly fixed matrix, $X \in R^{n\times n}$ is an unknown solution matrix, so most of the contents of the LMI solution are variable matrices. Formulas (1) and (2) are two representations of LMI, but they embody the same essence. Therefore, the resolution of a kind of LMI problem [24] is obtained as follows: for the existing LMI $G(x) < 0$, we can use the function that comes with LMI to solve it, in which there is a function with special meaning, which can be used to test whether the existing x satisfies $G(x) < 0$. This kind of case is the LMI technique to solve the problem, and the solver in the function is expressed by feasp.

2.2. H_∞ Control Theory. As the core part of robust control, H_∞, the theory provides many analysis ideas for control systems and effective solutions for robustness exploration. It can realize the calculation of H_∞ the norm. By optimizing this kind of function and then designing the controller reasonably within the prescribed constraint conditions, the robustness of the system becomes very good. The block diagram of H_∞ control is shown in Figure 1.

In addition, u is the output information of the controller, P is the controlled object, w is the external interference information, z is the information output of the object, y is the measured value, and K is the controller.

System objects are considered as follows:

$$\begin{cases} \dot{x}(t) = \overline{A}_{11}x(t) + \overline{B}_{11}u(t) + \overline{B}_{1w}w(t), \\ z(t) = \overline{C}_{22}x(t) + \overline{D}_{11}w(t) + \overline{D}_{12}u(t), \\ y(t) = \overline{C}_{33}x(t) + \overline{D}_{31}w(t) + \overline{D}_{32}u(t), \end{cases} \tag{3}$$

where $u(t) \in R^m$ is the input vector of the system, $y(t) \in R^s$ is the measured output of the system, $x(t) \in R^n$ denotes the state vector, $z(t) \in R^q$ represents the control output of the system, and $w(t) \in \mathscr{L}_2[0, \infty)$ represents the information of the disturbance signal. $\overline{A}_{11}, \overline{D}_{32}, \overline{B}_{11}, \overline{B}_{1w}, \overline{C}_{33}, \overline{D}_{21}, \overline{C}_{22}, \overline{D}_{11}, \overline{D}_{12}$ is a fixed constant matrix.

For any $\gamma > 0$, if formula (3) contains the following properties:

(1) If $w(t) = 0$, then the system is stable.

(2) If the function $\Phi(s)$ is used to represent the relationship between $z(t)$ and $w(t)$ by calculating the norm, we have

$$\|\Phi(s)\|_\infty = \sup_{\|w\|_2 \le 1} \frac{\|z\|_2}{\|w\|_2} \le \gamma^2. \tag{4}$$

And, then it is equal to

$$\int_0^\infty \gamma^2 w^T(t) w(t) \mathrm{d}t \ge \int_0^\infty z^T(t) z(t) \mathrm{d}t, \quad \forall w(t) \in \mathscr{L}_2[0, \infty). \tag{5}$$

Then, system (5) has H_∞ performance γ [25].

Formula (5) is the ability of anti-interference to the outside world, so γ is called the suppression system to the outside world. The smaller the value of γ is, the better the control performance of the system will be.

2.3. Some Lemmas

Lemma 1. *(Jensen inequality). For constant matrices $W \in R^{n\times n}$, $W = W^T \ge 0$, scalar $0 \le \tau_1 \le \tau_2$, and vector-valued functions $\dot{x}$: $[-\tau_2, -\tau_1] \longrightarrow R^n$. The following formula holds:*

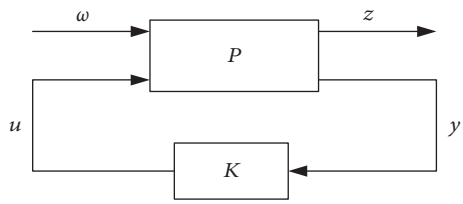


Figure 1: The structure of the H_∞ problem.

$$-(\tau_2-\tau_1)\int_{t-\tau_2}^{t-\tau_1}\dot{x}^T(s)W\dot{x}(s)\mathrm{d}s \le -\begin{bmatrix} x(t-\tau_1) \\ x(t-\tau_2)\end{bmatrix}^T \cdot \begin{bmatrix} W & -W \\ -W & W\end{bmatrix} \cdot \begin{bmatrix} x(t-\tau_1) \\ x(t-\tau_2)\end{bmatrix}. \tag{6}$$

Lemma 2. *(Schur complement lemma). For symmetric matrices* $W = \begin{bmatrix} W_1 & * \\ W_{12}^T & W_2\end{bmatrix}$, *where* W_{11} *is* $n \times n$ *dimensional, "*" is a symmetric term, and the following conditions are equivalent:* $W<0$, $W_1<0, W_2 - W_{12}^T W_1^{-1} W_{12}<0$, $W_2<0$, *and* $W_1 - W_{12}W_2^{-1}W_{12}^T<0$.

Lemma 3. *For the real matrix* $\Theta>0, X$ *and the given scalar* δ, *the following formula holds:* $\delta^2\Theta - 2\delta X \ge -X\Theta^{-1}X$.

3. Problem Description

In the current research results, most of the studies are completed under the assumption that the object can be measured. In the actual system, the information of the object cannot be measured directly, so the NCSs' study of state feedback under the DoS attack of event-triggered mechanism cannot act on the system directly. Therefore, the NCSs' research of output feedback under the DoS attack is critical.

3.1. System Description. The model of the system is as follows:

$$\begin{cases} \dot{x}_a(t) = (A_a + \Delta A_a(t))x_a(t) + B_a\hat{u}(t) + B_w w(t), \\ y(t) = C_a x_a(t), \\ z(t) = C_{a1}x_a(t) + D_a\hat{u}(t) + D_{a1}w(t), \\ x(t_0) = x_0, \end{cases} \tag{7}$$

where $\hat{u}(t) \in R^m$ is the input vector, $x_a(t) \in R^n$ is the state vector, $w(t) \in L_2[0,\infty)$ is the interference signal, $y(t) \in R^q$ is the measured output, and $A_a, B_a, C_a, C_{a_1}, D_a, B_w, D_{a1}$ is the constant matrix with appropriate dimension.

$\Delta A_a(t)$ matrix is an uncertain matrix and must satisfy the condition of (8):

$$\Delta A_a(t) = DF(t)E, \tag{8}$$

where D and E is a fixed constant matrix and $F(t)$ is a time-varying unknown matrix with respect to time t is Lebesque and satisfies $F^T(t)F(t) \le I$.

To improve the anti-interference ability of the system in the network, save the limited network resources, and apply the network model attacked by DoS to the control system, the system structure block diagram is shown in Figure 2.

3.2. Establishment of DoS Attack Model. Consider a general attack model that limits the actions of attackers only by limiting the frequency and duration of DoS attacks. It is a kind of periodic network interference signal with energy constraint in the form of pulse width modulation (PWM). Its specific expression is as follows:

$$P_{Dos}(t) = \begin{cases} 0, & t \in \left[(n-1)T, (n-1)T + T_{off}\right), \\ 1, & t \in \left[(n-1)T + T_{off}, nT\right), \end{cases} \tag{9}$$

where $n \in N$ is a periodic natural number, $T \in R_{>0}$ is an attack period, $T_{\text{off}} \in R_{>0}(T_{\text{off}} < T)$ is the zero boundary point of DoS attack and non-DoS attack, and $T_{\text{off}}^{\min} < T_{\text{off}} < T < \infty$. The interval of the network system that is not attacked by DoS is $[(n-1)T, (n-1)T + T_{\text{off}})$, and the interval of the network system that is attacked by DoS is $[(n-1)T + T_{\text{off}}, nT)$.

3.3. Establish a Double-Ended Elastic Event Trigger Mechanism. In view of the situation that the controlled object cannot be measured directly, we propose a two-terminal elastic event trigger mechanism which depends on the information of the object and the controller, which can enhance the anti-interference ability of the system and reduce the amount of data transmitted in the network channel at the same time.

The latest sensor trigger time is expressed by $b_k h$, and $b_{k+1}h$ is used to indicate the next trigger time. If there is no DoS attack, the elastic trigger conditions on the sensor side are

$$b_{k+1}h = b_k h + \min_{q\in N}\{qh | [y(b_k h) - y(b_k h + qh)]^T W_y \cdot [y(b_k h) - y(b_k h + qh)] \le \xi_y(t) y^T(b_k h) W_y y(b_k h)\}, \tag{10}$$

where $\xi_y(t)$ is used to represent the threshold function and $W_y > 0$ is used to represent the matrix that needs to be solved, and at the same time, it must satisfy

$$\dot{\xi}_y(t) = \frac{1}{\xi_y(t)}\left[\frac{1}{\xi_y(t)} - \delta_y\right][y(b_k h) - y(b_k h + qh)]^T W_y \cdot [y(b_k h) - y(b_k h + qh)], \tag{11}$$

where $\delta_y \ge 1$ is the known constant, $y(b_k h + qh)$ the sampling state of the current time, and $y(b_k h)$ shows the latest state of the event trigger.

If there is a DoS attack, the trigger condition of the sensor-side trigger will be defined as

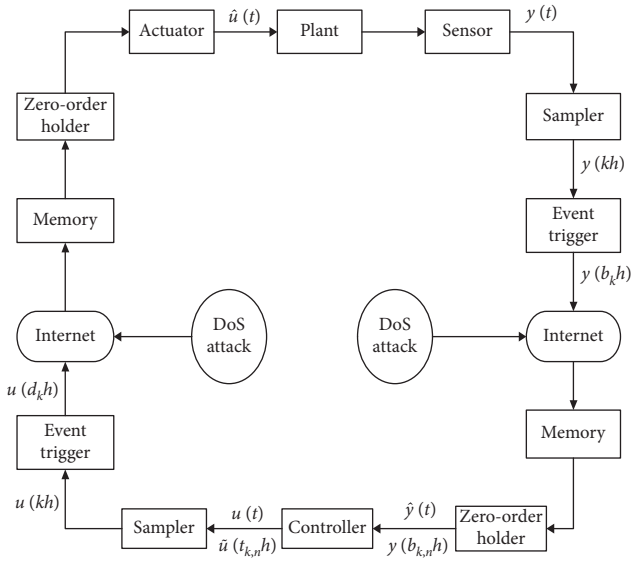


Figure 2: DoS attack NCS diagram of double-ended event triggering mechanism under output feedback.

$$b_{k,n}h = \left\{ b_{k_q} | b_{k_q} \in \left[(n-1)T,\ (n-1)T + T_{off}^{\min} \right) \right\} \cup \{nT\}. \tag{12}$$

If the designed elastic event trigger satisfies condition (10) or (12) and the current state $y(b_k h + qh)$ is defined as the latest state $y(b_{k+1}h)$, then it is sent to the network. $d_k h$ indicates the latest time of the controller, and $d_{k+1}h$ indicates the next trigger time. If there is no DoS attack, the elastic trigger conditions of the controller are as follows:

$$d_{k+1}h = d_k h + \min_{q \in N} \left\{ qh | \left[u(d_k h) - u(d_k h + qh) \right]^T W_u \cdot \left[u(d_k h) - u(d_k h + qh) \right] \le \xi_u(t) u^T(d_k h) W_u u(d_k h) \right\}. \tag{13}$$

Using $W_u > 0$ to express the matrix that needs to be solved, it must satisfy

$$\dot{\xi}_u(t) = \frac{1}{\xi_u(t)} \left[\frac{1}{\xi_u(t)} - \delta_u \right] \left[u(d_k h) - u(d_k h + qh) \right]^T W_u \cdot \left[u(d_k h) - u(d_k h + qh) \right]. \tag{14}$$

If there is a DoS attack, the event trigger conditions on the controller side need to be met as follows:

$$d_{k,n}h = \left\{ d_{k_q} | d_{k_q} \in \left[(n-1)T,\ (n-1)T + T_{off}^{\min} \right) \right\} \cup \{nT\}. \tag{15}$$

3.4. Establish the Model of Closed-Loop System. According to the study of timing and event mechanism in [25], we define $\lambda_{k,n} = b_{k+1} - b_k - 1$. According to the duration of ZOH, it is finally divided into segments h:

$$\left[t_{k,n}, t_{k+1,n} \right) h = \bigcup_{l_k=0}^{\lambda_{k,n}} Q_{k,n}^{l_k}. \tag{16}$$

In addition, $t_{0,n}h = (n-1)T\,(h < T)$ and $c(n) = \sup \left\{ k \in N | t_{k,} \quad nh \le (n-1)T + T_{off}^{\min} \right\}$. For $k \in c(n), n \in N$, $\lambda_{k,n} = \inf \left\{ m \in N | t_{k,n}h + mh \ge t_{k+1,n}h \ge nT \right\}$:

$$\begin{cases} C_{1,n} \widehat{=} \left[(n-1)T,\ (n-1)T + T_{off} \right),\ \vartheta(n) \widehat{=} \{0, 1, \ldots, k(n)\}, \\ C_{2,n} \widehat{=} \left[(n-1)T + T_{off}, nT \right),\ Q_{k,n}^{l_k} \widehat{=} \left[t_{k,n}, t_{k+1,n} \right) h, k \in \vartheta(n), \end{cases} \tag{17}$$

$$Q_{k,n}^{l_k} = \begin{cases} \left[t_{k,n}, t_{k,n} + h \right) h, & l_k = 0, \\ \left[t_{k,n} + h, t_{k,n} + 2h \right) h, & l_k = 1, \\ \vdots & \\ \left[t_{k,n} + (\lambda_k - 1)h, t_{k,n} + \lambda_k h \right) h, & l_k = \lambda_k - 1, \\ \left[t_{k,n} + \lambda_k h, t_{k+1,n} \right) h, & l_k = \lambda_k, \end{cases} \tag{18}$$

$$\begin{cases} \Upsilon_k^m = \left[t_{k,n}h + (m-1)h, t_{k,n}h + mh \right), \quad m = \left[1, 2, \ldots, \lambda_{k,n} - 1 \right], \\ \Upsilon_k^{\lambda_{k,n}} = \left[t_{k,n}h + (\lambda_{k,n} - 1)h, \quad t_{k+1,n}h \right), \end{cases} \tag{19}$$

$$C_{1,n} = \bigcup_{k=0}^{k(n)} \left\{ Q_{k,n}^{l_k} \cap C_{1,n} \right\} \subseteq \bigcup_{k=0}^{k(n)} Q_{k,n}^{l_k}. \tag{20}$$

The combination of (18)–(20) can be described as follows:

$$C_{1,n} = \bigcup_{k=0}^{k(n)} \bigcup_{m=1}^{\lambda_{k,n}} \left\{ \Upsilon_k^m \cap C_{1,n} \right\}. \tag{21}$$

Let $\psi_k^m = \Upsilon_k^m \cap C_{1,n}$, i.e., $C_{1,n} = \cup_{k=0}^{k(n)} \cup_{m=1}^{\lambda_{k,n}} \psi_k^m$.
Define the function, $\eta(t) = t - (b_{k,n}h + l_k h)$, $t \in \psi_k^m$, i.e.,

$$\eta(t) = \begin{cases} t - b_{k,n}h, & t \in \psi_k^1, \\ t - (b_{k,n}h + h), & t \in \psi_k^2, \\ \vdots & \\ t - \left(b_{k,n}h + \lambda_k h \right), & t \in \psi_k^{\lambda_k - 1}, \end{cases} \tag{22}$$

where $\tau \le \eta(1) \le \eta(t) \le \eta(2) = h + \tau$, $\dot{\eta}(t) = 1, \quad t \in \psi_k^m$.

The set of real-time update times of the feedback forward channel zero-order holder will be expressed as $\left\{ \tau, t_{1,n}, t_{2,n}, \ldots \right\}$, where $t_{k,n} = b_{k,n}h + \tau$. According to the relevant properties of ZOH, the input information of output feedback is

$$\hat{y}(t) = y\left(b_{k,n}h \right), \quad t \in \left[t_{k,n}, t_{k+1,n} \right) h. \tag{23}$$

The control input of the controlled plant (7) is

$$\hat{u}(t) = u\left(\overline{d}_{k,n}h \right), \quad t \in \psi_k^m. \tag{24}$$

Several functions are defined as follows.

(1) Define the function:

$$e_y(t) = y\left(b_{k,n}h\right) - y\left(b_{k,n}h + l_k h\right), \quad t \in \psi_k^m. \tag{25}$$

According to formulas (22), (23), and (25), the controller inputs as

$$\widehat{y}(t) = y\left(b_{k,n}h\right) = \left[e_y(t) + y(t-\eta(t))\right], \quad t \in \psi_k^m. \tag{26}$$

(2) Define the function:

$$e_u(t) = u\left(\overline{d}_{k,n}h\right) - u\left(b_{k,n}h + l_k h\right), \quad t \in \psi_k^m. \tag{27}$$

According to formulas (22), (24), and (25), the control input of the controlled object (7) is

$$\widehat{u}(t) = u\left(\overline{d}_{k,n}h\right) = e_u(t) + u(t-\eta(t)), \quad t \in \psi_k^m. \tag{28}$$

The feedback controller for output dynamics is

$$\begin{cases} \dot{x}(t) = A_c x_c(t) + A_{cd} x_c(t-\eta(t)) + B_c \widehat{y}(t), \\ u(t) = K x_c(t), \quad t \in \psi_k^m, \end{cases} \tag{29}$$

where $\widehat{y}(t)$ is the input vector of dynamic output feedback, $u(t)$ is the output vector, and $x_c(t) \in R^n$ matrix is the matrix with a proper dimension of the state vector A_c, A_{cd}, K, B_c.

In the same timing, the input $\widehat{u}(t)$ of (29) is substituted into model (7), and the input $\widehat{y}(t)$ of (26) is brought into (29). The final closed-loop model is

$$\begin{cases} \begin{cases} \dot{\varepsilon}(t) = A_1\varepsilon(t) + A_{12}\varepsilon(t-\eta(t)) + B_{11}e_y(t) + B_{12}e_u(t) + \overline{B}_w w(t), \\ z(t) = \overline{C}_{a1}\varepsilon(t) + D_a e_u(t) + D_a K\varepsilon(t-\eta(t)) + D_{a1}w(t), \quad t \in Q_{k,n}^{l_k} \cap C_{1,n}, \end{cases} \\ \begin{cases} \dot{\varepsilon}(t) = A_1\varepsilon(t) + A_{22}\varepsilon(t-\eta(t)) + B_{11}e_y(t) + \overline{B}_w w(t), \\ z(t) = \overline{C}_{a1}\varepsilon(t) + D_{a1}w(t), \quad t \in C_{2,n}. \end{cases} \end{cases} \tag{30}$$

In order to facilitate the analysis, this chapter introduces the switching signal:

$$\psi(t) = \begin{cases} 1, & t \in [-h, 0) \cup \left(\underset{n\in N}{\cup} C_{1,n}\right), \\ 2, & t \in \underset{n\in N}{\cup} C_{2,n}. \end{cases} \tag{31}$$

For $\psi(t) = i \in \{1, 2\}$ and$n \in N$, define

$$t_{i,n} = \begin{cases} (n-1)T, & i = 1, \\ (n-1)T + T_{off}^{\min}, & i = 2. \end{cases} \tag{32}$$

Let $C_{i,n} = [t_{i,n}, t_{3-i,n+i-1}), \quad \psi(t_{i,n}) = i, \psi(t_{i,n}^-) = 3 - i.$

Based on the set switching signal $\psi(t)$, the closed-loop system (30) can be simplified as follows:

$$\begin{cases} \dot{\varepsilon}_{\psi(t)}(t) = A_1\varepsilon(t) + A_{i2}\varepsilon(t-\eta(t)) + B_{11}e_y(t) + B_{i2}e_u(t) + \overline{B}_w w(t), \\ i \in \{1, 2\}, \quad t \in \underset{n\in N}{\cup}\left[t_{i,n}, t_{3-i,n+i-1}\right), \\ z(t) = \overline{C}_{a1}\varepsilon(t) + D_{a1}w(t) + D_{ia}e_u(t) + D_{ia}K\varepsilon(t-\eta(t)), \\ x(t_0) = x_0, \end{cases} \tag{33}$$

where

$$\begin{aligned} D_{1a} &= D_a, \\ D_{2a} &= 0, \\ B_{22} &= 0, \\ \overline{C}_{a1} &= \begin{bmatrix} C_{a1} & 0 \end{bmatrix}, \\ \varepsilon(t) &= \begin{bmatrix} x_a(t) \\ x_c(t) \end{bmatrix}, \\ \overline{B}_w &= \begin{bmatrix} B_w \\ 0 \end{bmatrix}, \\ B_{11} &= \begin{bmatrix} 0 \\ B_c \end{bmatrix}, \\ B_{12} &= \begin{bmatrix} B_a \\ 0 \end{bmatrix}, \\ A_1 &= \begin{bmatrix} A_a + \Delta A_a(t) & 0 \\ 0 & A_c \end{bmatrix}, \\ A_{12} &= \begin{bmatrix} 0 & B_a K \\ B_c C_a & A_{cd} \end{bmatrix}, \\ A_{22} &= \begin{bmatrix} 0 & 0 \\ B_c C_a & A_{cd} \end{bmatrix}. \end{aligned} \tag{34}$$

4. Stability Analysis of Closed-Loop System

Through the LMI method and the related theory of Lyapunov, aiming at whether the networked control system is attacked by DoS or not in the unmeasurable state, based on the proposed two-terminal elastic event trigger mechanism, the exponential stability of the system is proved.

Theorem 1. *For a given state gain matrix K and interference signal $P_{DoS}(t)$ (9), the parameters T and $T_{off}^{\min}$ are known. Consider system (33), for the known positive scalar $\alpha_i, h > 0$ and $\gamma > 0$ and the double-ended elastic event trigger parameter $0 < \delta_y < 1$ and $0 < \delta_u < 1$, if there is a positive definite matrix W_u, W_y, P_i, Q_i, R_i, and Z_i and the matrix M_i, N_i, S_i, $i \in \{1, 2\}$, while satisfying the following conditions*

$$\sum_i = \begin{bmatrix} \Pi_{11}^i & * & * & * & * & * \\ \sqrt{h}N_i^T & \Upsilon_{22}^i & * & * & * & * \\ \sqrt{h}S_i^T & 0 & \Upsilon_{33}^i & * & * & * \\ \sqrt{h}M_i^T & 0 & 0 & \Upsilon_{44}^i & * & * \\ \sqrt{h}R_i\overline{A}_i & 0 & 0 & 0 & \Upsilon_{55}^i & * \\ \sqrt{h}Z_i\overline{A}_i & 0 & 0 & 0 & 0 & \Upsilon_{66}^i \end{bmatrix} < 0, \quad i = 1, 2,$$

$$\begin{aligned} \Pi_{11}^i &= \Phi_{i1} + \Phi_i + \Phi_i^T, \\ \Upsilon_{22}^i &= \Upsilon_{33}^i \\ \Upsilon_{44}^i &= -e^{2(-1)^i\alpha_i\lambda_i h}Z_i, \\ \lambda_1 &= 1, \\ \lambda_2 &= 0, \\ \Upsilon_{55}^i &= -R_i, \\ \Upsilon_{66}^i &= -Z_i, \end{aligned} \tag{35}$$

then along the system trajectories (4)–(21), for any $n \in N$, there are

$$V_i(t) \le e^{2(-1)^i\alpha_i(t-t_{i,n})}V_i(t_{i,n}), \quad t \in [t_{i,n}, t_{3-i,n+i-1}). \tag{36}$$

Proof. Select the following time-varying Lyapunov functions:

$$\begin{aligned} V_{\psi(t)}(t) = {} & \varepsilon^T(t)P_{\psi(t)}\varepsilon(t) + \int_{t-h}^{t}\varepsilon^T(s)\exp(\bullet)Q_{\psi(t)}\varepsilon(s)\mathrm{d}s \\ & + \int_{-h}^{0}\int_{t+\theta}^{t}\dot{\varepsilon}^T(s)\exp(\bullet)Z_{\psi(t)}\dot{\varepsilon}(s)\mathrm{d}s\mathrm{d}\theta \\ & + \int_{-h}^{0}\int_{t+\theta}^{t}\dot{\varepsilon}^T(s)\exp(\bullet)R_{\psi(t)}\dot{\varepsilon}(s)\mathrm{d}s\mathrm{d}\theta \\ & + \frac{1}{2}\xi_y^2(t) + \frac{1}{2}\xi_u^2(t). \end{aligned} \tag{37}$$

When $P_{\psi(t)} > 0, R_{\psi(t)} > 0, Q_{\psi(t)} > 0, Z_{\psi(t)} > 0, \alpha_{\psi(t)} > 0$, and $\exp(\bullet) = e^{2(-1)\psi(t)\alpha_{\psi(t)}(t-s)}$ are given, for any $k \in \vartheta(n)$, $n \in N$, when $\psi(t) = 1$, calculate the derivative $V_1(t)$ at $t \in Q_{k,n}^l \cap C_{1,n}$:

$$\begin{aligned} \dot{V}_1(t) \le {} & -2\alpha_1 V_1(t) + 2\alpha_1\varepsilon^T(t)P_1\varepsilon(t) + 2\varepsilon^T(t)P_1\dot{\varepsilon}(t) + \varepsilon^T(t)Q_1\varepsilon(t) \\ & - \varepsilon^T(t-h)e^{-2\alpha_1 h}Q_1\varepsilon(t-h) + h\dot{\varepsilon}^T(t)(R_1 + Z_1)\dot{\varepsilon}(t) \\ & - \int_{t-h}^{t}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}Z_1\dot{\varepsilon}(s)\mathrm{d}s - \int_{t-\eta(t)}^{t}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}R_1\dot{\varepsilon}(s)\mathrm{d}s \\ & + 2\tilde{\varepsilon}^T(t)M_1\left[\varepsilon(t) - \varepsilon(t-h) - \int_{t-h}^{t}\dot{\varepsilon}(s)\mathrm{d}s\right] \\ & + 2\tilde{\varepsilon}^T(t)N_1\left[\varepsilon(t) - \varepsilon(t-\eta(t)) - \int_{t-\eta(t)}^{t}\dot{\varepsilon}(s)\mathrm{d}s\right] \\ & + 2\tilde{\varepsilon}^T(t)S_1\left[\varepsilon(t-\eta(t)) - \varepsilon(t-h) - \int_{t-h}^{t-\eta(t)}\dot{\varepsilon}(s)\mathrm{d}s\right] \\ & - \int_{t-h}^{t-\eta(t)}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}R_1\dot{\varepsilon}(s)\mathrm{d}s + \xi_y(t)\dot{\xi}_y(t) + \xi_u(t)\dot{\xi}_u(t), \quad t \in Q_{k,n}^l \cap C_{1,n}. \end{aligned} \tag{38}$$

According to the literature [25], we can obtain

$$-2\tilde{\varepsilon}^T(t)M_1\int_{t-h}^{t}\dot{\varepsilon}(s)\mathrm{d}s \le h\tilde{\varepsilon}^T(t)M_1e^{2\alpha_1 h}Z_1^{-1}M_1^T\tilde{\varepsilon}(t)+\int_{t-h}^{t}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}Z_1\dot{\varepsilon}(s)\mathrm{d}s, \tag{39}$$

$$-2\tilde{\varepsilon}^T(t)N_1\int_{t-\eta(t)}^{t}\dot{\varepsilon}(s)\mathrm{d}s \le h\tilde{\varepsilon}^T(t)N_1e^{2\alpha_1 h}R_1^{-1}N_1^T\tilde{\varepsilon}(t)+\int_{t-\eta(t)}^{t}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}R_1\dot{\varepsilon}(s)\mathrm{d}s, \tag{40}$$

$$-2\tilde{\varepsilon}^T(t)S_1\int_{t-h}^{t-\eta(t)}\dot{\varepsilon}(s)\mathrm{d}s \le h\tilde{\varepsilon}^T(t)S_1e^{2\alpha_1 h}R_1^{-1}S_1^T\tilde{\varepsilon}(t)+\int_{t-h}^{t-\eta(t)}\dot{\varepsilon}^T(s)e^{-2\alpha_1 h}R_1\dot{\varepsilon}(s)\mathrm{d}s. \tag{41}$$

The simplified expressions of the two-terminal elastic event-triggering mechanisms (10)–(13) and trigger functions (23) and (28) can be obtained from the same time sequence ψ_k^m of the input $\hat{y}(t)$ of equation (23) and the object input $\hat{u}(t)$ of (28):

$$\frac{1}{\xi_y(t)}e_y^T(t)W_ye_y(t) \le \left[e_y(t)+y(t-\eta(t))\right]^T W_y \cdot \left[e_y(t)+y(t-\eta(t))\right], \quad t\in Q_{k,n}^{l_k}\cap C_{1,n}, \tag{42}$$

$$\begin{aligned}\xi_y(t)\dot{\xi}_y(t) &= \left[\frac{1}{\xi_y(t)}-\delta_y\right]e_y^T(t)W_ye_y(t)\\ &= \frac{1}{\xi_y(t)}e_y^T(t)W_ye_y(t)-\delta_ye_y^T(t)W_ye_y(t)\\ &\le \left[e_y(t)+y(t-\eta(t))\right]^T W_y\left[e_y(t)+y(t-\eta(t))\right]\\ &\quad -\delta_ye_y^T(t)W_ye_y(t), \quad t\in Q_{k,n}^{l_k}\cap C_{1,n},\end{aligned} \tag{43}$$

$$\frac{1}{\xi_u(t)}e_u^T(t)W_ue_u(t) \le \left[e_u(t)+u(t-\eta(t))\right]^T W_u \cdot \left[e_u(t)+u(t-\eta(t))\right], \quad t\in Q_{k,n}^{l_k}\cap C_{1,n}, \tag{44}$$

$$\begin{aligned}\xi_u(t)\dot{\xi}_u(t) &= \left[\frac{1}{\xi_u(t)}-\delta_u\right]e_u^T(t)W_ue_u(t)\\ &= \frac{1}{\xi_u(t)}e_u^T(t)W_ue_u(t)-\delta_ue_u^T(t)W_ue_u(t)\\ &\le \left[e_u(t)+u(t-\eta(t))\right]^T W_u\left[e_u(t)+u(t-\eta(t))\right]\\ &\quad -\delta_ue_u^T(t)W_ue_u(t), \quad t\in Q_{k,n}^{l_k}\cap C_{1,n}.\end{aligned} \tag{45}$$

Combining formulas (38)–(41), (43), and (45),

$$\begin{aligned}\dot{V}_1(t) \le & -2\alpha_1V_1(t)+\chi^T(t)\Big[\Pi_{111}^1+hN_1e^{2\alpha_1 h}R_1^{-1}N_1^T\\ &+hM_1e^{2\alpha_1 h}Z_1^{-1}M_1^T+hS_1e^{2\alpha_1 h}R_1^{-1}S_1^T\\ &+h\overline{A}_1^T(R_1+Z_1)\overline{A}_1\Big]\chi(t),\end{aligned} \tag{46}$$

where $\chi(t)=\left\{\varepsilon(t),\varepsilon(t-\eta(t)),\varepsilon(t-h),e_y(t),e_u(t),w(t)\right\}$.

Apparently,

$$\begin{aligned}\dot{V}_1(t) &= \dot{V}_1(t)+\gamma^2w(t)w^T(t)-\gamma^2w(t)w^T(t)\\ &\quad -z(t)z^T(t)+z(t)z^T(t).\end{aligned} \tag{47}$$

$z(t)$ of (29) in the closed-loop system is brought into (47) to obtain (48):

$$\begin{aligned}\dot{V}_1(t) \le & -2\alpha_1V_1(t)+\chi^T(t)\Big[\Pi_{11}^1+hN_1e^{2\alpha_1 h}R_1^{-1}N_1^T+hM_1e^{2\alpha_1 h}Z_1^{-1}M_1^T+hS_1e^{2\alpha_1 h}R_1^{-1}S_1^T+h\overline{A}_1^T(R_1+Z_1)\overline{A}_1\Big]\chi\\ &\cdot(t)+\gamma^2w^T(t)w(t)-z^T(t)z(t),\end{aligned}$$

$$\Phi_{11}=\begin{bmatrix} P_{11} & * & * & * & * & *\\ A_{12}^TP_1+K^TD_a^TC_{a1} & H_{221} & * & * & * & *\\ 0 & 0 & -e^{-2\alpha_1 h}Q_1 & * & * & *\\ B_{11}^TP_1 & W_yC_a & 0 & W_y-\delta_yW_y & * & *\\ D_a^TC_{a1}+B_{12}^TP_1 & H_{52} & 0 & 0 & H_{551} & *\\ \overline{B}_w^TP_1+D_{a1}^TC_{a1} & D_{a1}^TD_aK & 0 & 0 & D_{a1}^TD_a & D_{a1}^TD_{a1}-\gamma^2 \end{bmatrix}, \tag{48}$$

$$P_{11}=P_1A_1+A_1^TP_1+Q_1+2\alpha_1P_1+C_{a1}^T*C_{a1}, H_{551}=-\delta_uW_u+W_u+D_a^TD_a,$$

$$\Phi_1=\begin{bmatrix} M_1+N_1 & -N_1+S_1 & -M_1-S_1 & 0 & 0 & 0\end{bmatrix},$$

$$\overline{A}_1=\begin{bmatrix} A_1 & A_{12} & 0 & B_{11} & B_{12} & \overline{B}_w\end{bmatrix},$$

$$H_{221}=K^TW_uK+C_a^TW_yC_a+K^TD_a^TD_aK, H_{52}=W_uK+D_a^TD_aK.$$

By using Shur Lemma, the following results are obtained:

$$\begin{aligned}&\Pi_{11}^{1}+hN_1e^{2\alpha_1 h}R_1^{-1}N_1^T+hM_1e^{2\alpha_1 h}Z_1^{-1}M_1^T\\&\quad+hS_1e^{2\alpha_1 h}R_1^{-1}S_1^T+h\overline{A}_1^T(R_1+Z_1)\overline{A}_1<0.\end{aligned}\tag{49}$$

On the contrary, when $t\in[t_{2,n},t_{1,n+1})$, the same result is obtained:

$$\begin{aligned}\dot{V}_2(t)&\le 2\alpha_2V_2(t)+\chi^T(t)\Big[\Pi_{11}^{2}+hN_2R_2^{-1}N_2^T+hM_2Z_2^{-1}M_2^T+hS_2R_2^{-1}S_2^T+hS_2R_2^{-1}S_2^T+h\overline{A}_2^T(R_2+Z_2)\overline{A}_2\Big]\chi\\&\quad\cdot(t)+\gamma^2w^T(t)w(t)-z^T(t)z(t),\\\Phi_{21}&=\begin{bmatrix}P_2A_1+A_1^TP_2+2\alpha_2P_2+Q_2+C_{a1}^T*C_{a1} & * & * & * & * & *\\A_{22}^TP_2 & H_{222} & * & * & * & *\\0 & 0 & -Q_2 & * & * & *\\B_{11}^TP_2 & W_yC_a & 0 & W_y-\delta_yW_y & * & *\\0 & W_uK & 0 & 0 & H_{552} & *\\\overline{B}_w^TP_2+D_{a1}^TC_{a1} & 0 & 0 & 0 & 0 & D_{a1}^TD_{a1}-\gamma^2\end{bmatrix},\\H_{222}&=K^TW_uK+C_a^TW_yC_a,\\H_{552}&=-\delta_uW_u+W_u,\\\overline{A}_2&=\begin{bmatrix}A_1 & A_{22} & 0 & B_{11} & 0 & \overline{B}_w\end{bmatrix},\\\Phi_2&=\begin{bmatrix}M_2+N_2 & -N_2+S_2 & -M_2-S_2 & 0 & 0 & 0\end{bmatrix}.\end{aligned}\tag{50}$$

According to the condition $\sum_2<0$, it is proved that it is completed if it satisfies $\dot{V}_2(t)\le 2\alpha_2V_2(t)$.

Theorem 2. *For a given state gain matrix K and interference signal $P_{DoS}(t)$ (9), consider system (33), when the sequence satisfies$\{nT\}_{n\in N}$, and parameters T and $T_{off}^{\min}$ are known. For known positive scalars $\alpha_i,\mu_i,h>0$ and $\gamma>0$ and the trigger parameter $0<\delta_y<1, 0<\delta_u<1$, such as symmetric, positive, definite matrix, W_y,W_u,P_i,Q_i,R_i, and Z_i, and matrix $M_i,N_i,S_i,i\in\{1,2\}$. Both (35) and the following inequalities are satisfied:*

$$\begin{aligned}&Q_i\le\mu_{3-i}Q_{3-i},\\&P_1\le\mu_2P_2,\\&Z_i\le\mu_{3-i}Z_{3-i},\\&P_2\le\mu_1e^{2(\alpha_1+\alpha_2)h}P_1,\\&R_i\le\mu_{3-i}R_{3-i},\\&0<\lambda=-2(\alpha_1+\alpha_2)h+2\alpha_1T_{\text{off}}^{\min}-\text{In}(\mu_1\mu_2)\\&\quad-2\alpha_2\left(T-T_{\text{off}}^{\min}\right).\end{aligned}\tag{51}$$

The closed-loop system (33) under known periodic DoS interference attacks (9) is globally exponentially stable with a decay rate.

According to the judgment basis of Theorems 1 and 2, it is determined that the system is globally exponentially stable. However, there is a coupling relationship between the corresponding matrix (A_c,A_{cd},K,B_c) and P_i in formula (29), so the controller cannot be directly designed. Therefore, Section 5 is needed to design the controller further.

5. Design of H_∞ Dynamic Output Feedback Controller

Theorem 3. *For the given matrix K and pulse width interference signal $P_{DoS}(t)$ (9), the sequence $\{nT\}_{n\in N}$ and parameters T and $T_{off}^{\min}$ are known. Considering series (33), for the known positive scalar, $\alpha_i,h>0,\gamma>0,\gamma_n(n=1,2,3,4,5)$ and, double-ended elastic event trigger parameter, $0<\delta_y<1$ and $0<\delta_u<1$. If it exists in the positive definite matrix, $W_u,W_y,\widetilde{P}_i,\widetilde{Q}_i,\widetilde{R}_i,\widetilde{Z}_i,X$, and Y and the matrix $\widetilde{M}_i,\widetilde{N}_i,\widetilde{S}_i$, $i\in\{1,2\}$, $Z=\begin{bmatrix}X & *\\I & Y\end{bmatrix}>0$, which is satisfied at the same time:*

$$\widetilde{\Theta}_1=\begin{bmatrix}\widetilde{\Pi}_{11}^i & * & * & * & * & *\\\sqrt{h}\widetilde{N}_i^T & \widetilde{\Upsilon}_{22}^i & * & * & * & *\\\sqrt{h}\widetilde{S}_i^T & 0 & \widetilde{\Upsilon}_{33}^i & * & * & *\\\sqrt{h}\widetilde{M}_i^T & 0 & 0 & \widetilde{\Upsilon}_{44}^i & * & *\\\sqrt{h}\widetilde{R}_i\widetilde{A}_i & 0 & 0 & 0 & \widetilde{\Upsilon}_{55}^i & *\\\sqrt{h}\widetilde{Z}_i\widetilde{A}_i & 0 & 0 & 0 & 0 & \widetilde{\Upsilon}_{66}^i\end{bmatrix}<0,\quad i=1,2.\tag{52}$$

Then, the closed-loop system (33) is globally exponentially stable under the double-ended elastic event trigger

mechanism (11) to (23). Also, the gain matrix of the output feedback controller (29) can be calculated as follows:

$$\begin{aligned} A_c &= N^{-1}\left(\Lambda_2 - YA_aX\right)(I - YX)^{-1}N, \\ B_c &= N^{-1}\Lambda_1, \\ K &= \Lambda_3(I - YX)^{-1}N, \\ A_{cd} &= N^{-1}\left(\Lambda_4 - \Lambda_1 C_a X - YB_a\Lambda_3\right)(I - YX)^{-1}N. \end{aligned} \tag{53}$$

Proof. The matrix P is decomposed into

$$P_i = \begin{bmatrix} Y & * \\ N^T & N^T\left(Y - X^{-1}\right)^{-1}N \end{bmatrix}, \quad i = 1, 2, \tag{54}$$

using Lemma 1

$$P_i > 0 \Leftrightarrow Z = \begin{bmatrix} X & * \\ I & Y \end{bmatrix} > 0 \Leftrightarrow \begin{cases} Y - X^{-1} > 0, \\ X > 0. \end{cases} \tag{55}$$

Define the matrix:

$$\begin{aligned} \Upsilon_1 &= \begin{bmatrix} X & I \\ N^{-1}(I - YX) & 0 \end{bmatrix}, \\ \Upsilon_2 &= P_i\Upsilon_1 = \begin{bmatrix} I & Y \\ 0 & N^T \end{bmatrix}, \\ \Omega &= \operatorname{diag}\{\Upsilon_1, \Upsilon_1, \Upsilon_1, \Upsilon_1, I, I, \Upsilon_2, \Upsilon_2, \Upsilon_2, \Upsilon_2, \Upsilon_2\}. \end{aligned} \tag{56}$$

By using Ω to perform the congruent transformation of the conditions in Theorem 1, the result is obtained:

$$\widetilde{\Theta}_1 = \Omega^T\Theta_1\Omega < 0, \quad i = 1, 2,$$

$$\widetilde{\Theta}_1 = \begin{bmatrix} \widetilde{\Pi}_{11}^i & * & * & * & * & * \\ \sqrt{h}\,\widetilde{N}_i^T & \widetilde{\Upsilon}_{22}^i & * & * & * & * \\ \sqrt{h}\,\widetilde{S}_i^T & 0 & \widetilde{\Upsilon}_{33}^i & * & * & * \\ \sqrt{h}\,\widetilde{M}_i^T & 0 & 0 & \widetilde{\Upsilon}_{44}^i & * & * \\ \sqrt{h}\,\widetilde{R}_i\widetilde{A}_i & 0 & 0 & 0 & \widetilde{\Upsilon}_{55}^i & * \\ \sqrt{h}\,\widetilde{Z}_i\widetilde{A}_i & 0 & 0 & 0 & 0 & \widetilde{\Upsilon}_{66}^i \end{bmatrix},$$

$$\Theta_1 = \begin{bmatrix} \Pi_{11}^i & * & * & * & * & * \\ \sqrt{h}\,N_i^T & \Upsilon_{22}^i & * & * & * & * \\ \sqrt{h}\,S_i^T & 0 & \Upsilon_{33}^i & * & * & * \\ \sqrt{h}\,M_i^T & 0 & 0 & \Upsilon_{44}^i & * & * \\ \sqrt{h}\,R_i\overline{A}_i & 0 & 0 & 0 & \Upsilon_{55}^i & * \\ \sqrt{h}\,Z_i\overline{A}_i & 0 & 0 & 0 & 0 & \Upsilon_{66}^i \end{bmatrix},$$

$$\begin{aligned} \widetilde{\Pi}_{11}^i &= \widetilde{\Phi}_{i1} + \widetilde{\Phi}_i + \widetilde{\Phi}_i^T, \\ \widetilde{\Upsilon}_{22}^i &= \widetilde{\Upsilon}_{33}^i = -Ze^{2(-1)^i\alpha_i\lambda_i h}\widetilde{R}_i Z, \\ \Upsilon_{44}^i &= -Ze^{2(-1)^i\alpha_i\lambda_i h}\widetilde{Z}_i Z, \\ \lambda_1 &= 1, \\ \lambda_2 &= 0, \end{aligned}$$

$$\widetilde{\Phi}_{11} = \begin{bmatrix} \widetilde{P}_{11} & * & * & * & * & * \\ \Upsilon_1^T A_{12}^T P_1\Upsilon_1 + \Upsilon_1^T K^T D_a^T C_{a1}\Upsilon_1 & \Upsilon_1^T H_{221}\Upsilon_1 & * & * & * & * \\ 0 & 0 & -e^{-2\alpha_1 h}\widetilde{Q}_1 & * & * & * \\ B_{11}^T P_1\Upsilon_1 & \Upsilon_1^T W_y C_a\Upsilon_1 & 0 & \Upsilon_1^T\left(W_y - \delta_y W_y\right)\Upsilon_1 & * & * \\ D_a^T C_{a1}\Upsilon_1 + B_{12}^T P_1\Upsilon_1 & H_{52}\Upsilon_1 & 0 & 0 & H_{551} & * \\ \overline{B}_w^T P_1\Upsilon_1 + D_{a1}^T C_{a1}\Upsilon_1 & D_{a1}^T D_a K\Upsilon_1 & 0 & 0 & D_{a1}^T D_a & D_{a1}^T D_{a1} - \gamma^2 \end{bmatrix},$$

$$
\begin{aligned}
\widetilde{P}_{11} &= \Upsilon_1^T\left(P_1A_1 + A_1^TP_1 + 2\alpha_1P_1 + C_{a1}^T * C_{a1}\right)\Upsilon_1 + \widetilde{Q}_1, \widetilde{\Upsilon}_{55}^i = -\widetilde{R}_i,\\
\widetilde{\Upsilon}_{66}^i &= -\widetilde{Z}_i, \widetilde{Z}_i = \Upsilon_2^TZ_i\Upsilon_2, \widetilde{Q}_i = \Upsilon_1^TQ_i\Upsilon_1,\\
\widetilde{N}_i &= \Upsilon_2^TN_i\Upsilon_2,\\
\widetilde{M}_i &= \Upsilon_2^TM_i\Upsilon_2,\\
\widetilde{S}_i &= \Upsilon_2^TS_i\Upsilon_2,\\
H_{551} &= \delta_uW_u + W_u + D_a^TD_a,\\
\widetilde{R}_i &= \Upsilon_2^TR_i\Upsilon_2,\\
H_{221} &= K^TW_uK + C_a^TW_yC_a + K^TD_a^TD_aK,\\
H_{52} &= W_uK + D_a^TD_aK,\\
\widetilde{\Phi}_i &= \left[\begin{array}{cccccc}\widetilde{M}_i + \widetilde{N}_i & -\widetilde{N}_i + \widetilde{S}_i & -\widetilde{M}_i - \widetilde{S}_i & 0 & 0 & 0\end{array}\right],\\
\widetilde{A}_1 &= \left[\begin{array}{cccccc}A_1 & A_{12} & 0 & B_{11} & B_{12} & \overline{B}_w\end{array}\right],\\
\widetilde{\Phi}_{21} &= \begin{bmatrix}
\widetilde{P}_{22} & * & * & * & * & *\\
\Upsilon_1^TA_{22}^TP_2\Upsilon_1 & \Upsilon_1^TH_{222}\Upsilon_1 & * & * & * & *\\
0 & 0 & -\widetilde{Q}_2 & * & * & *\\
B_{11}^TP_2\Upsilon_1 & \Upsilon_1^TW_yC_a\Upsilon_1 & 0 & \Upsilon_1^T\left(W_y - \delta_yW_y\right)\Upsilon_1 & * & *\\
0 & W_uK\Upsilon_1 & 0 & 0 & H_{552} & *\\
\overline{B}_w^TP_2\Upsilon_1 + D_{a1}^TC_{a1}\Upsilon_1 & 0 & 0 & 0 & 0 & D_{a1}^TD_{a1} - \gamma^2
\end{bmatrix},\\
\widetilde{A}_2 &= \left[\begin{array}{cccccc}A_1 & A_{12} & 0 & B_{11} & 0 & \overline{B}_w\end{array}\right],\\
H_{552} &= -\delta_uW_u + W_u,\\
H_{222} &= K^TW_uK + C_a^TW_yC_a,\\
\widetilde{P}_{22} &= \Upsilon_1^T\left(P_2A_1 + A_1^TP_2 + 2\alpha_2P_2 + C_{a1}^T * C_{a1}\right)\Upsilon_1 + \widetilde{Q}_2.
\end{aligned}
\tag{57}
$$

For conversions,

$$
\begin{aligned}
\Lambda_1 &= NB_c,\\
\Lambda_2 &= YA_aX + NA_cN^{-1}(I - YX),\\
\Lambda_3 &= KN^{-1}(I - YX),\\
\Lambda_4 &= \Lambda_1C_aX + YB_a\Lambda_3 + NA_{cd}N^{-1}(I - YX).
\end{aligned}
\tag{58}
$$

Applying Lemma 3 to deal with the coupling term in $\widetilde{\Upsilon}_{22}^i$, $\widetilde{\Upsilon}_{33}^i$, and $\widetilde{\Upsilon}_{44}^i$, the result is obtained:

$$
\operatorname{diag}\left\{\widetilde{\Upsilon}_{22}^i, \widetilde{\Upsilon}_{33}^i, \widetilde{\Upsilon}_{44}^i\right\} = \operatorname{diag}\left\{\gamma_1^2\widetilde{R}_i - 2\gamma_1Z, \gamma_2^2\widetilde{R}_i - 2\gamma_2Z, \gamma_3^2\widetilde{Z}_i - 2\gamma_3Z\right\}.
\tag{59}
$$

The gain matrix of the controller with output feedback is

$$
\begin{aligned}
A_c &= N^{-1}\left(\Lambda_2 - YA_aX\right)(I - YX)^{-1}N,\\
B_c &= N^{-1}\Lambda_1,\\
K &= \Lambda_3(I - YX)^{-1}N,\\
A_{cd} &= N^{-1}\left(\Lambda_4 - \Lambda_1C_aX - YB_a\Lambda_3\right)(I - YX)^{-1}N.
\end{aligned}
\tag{60}
$$

Because equation (59) contains an unknown matrix N, the gain (A_c, A_{cd}, K, B_c) cannot be solved directly. Therefore, through the transformation of $x_c(t) = N^{-1}\overline{x}_c(t)$, the equivalent of solution (29) is as follows:

$$
\begin{cases}
\dot{\overline{x}}_c(t) = \overline{A}_c\overline{x}_c(t) + \overline{D}_c\overline{x}_c(t - \eta(t)) + \overline{B}_c\overline{y}_p(t),\\
u(t) = \overline{K}x_c(t), \quad t \in \psi_k^m.
\end{cases}
\tag{61}
$$

The gain of the controller is expressed as follows:

$$
\begin{aligned}
\overline{A}_c &= \left(\Lambda_2 - YA_aX\right)(I - YX)^{-1},\\
\overline{B}_c &= \Lambda_1,\\
\overline{K} &= \Lambda_3(I - YX)^{-1},\\
\overline{A}_{cd} &= \left(\Lambda_4 - \Lambda_1C_aX - YB_a\Lambda_3\right)(I - YX)^{-1}.
\end{aligned}
\tag{62}
$$

6. Example Simulation

Considering the general satellite system model in [26, 27] as an example, the specific state-space model is described as (7), and the described matrix is given:

$$
\begin{aligned}
A_a &= \begin{bmatrix} 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 1.0000 \\ -0.28 & 0.28 & -0.0043 & 0.0043 \\ 0.28 & -0.28 & 0.0043 & -0.0043 \end{bmatrix}, \\
\Delta A_a &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10/3 & 0 \\ 0 & 0 & 0 & 10/3 \end{bmatrix}, \\
B_a &= \begin{bmatrix} 0 & 0 & 1 & 0.2178 \end{bmatrix}^T, \\
B_w &= \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}^T, \\
C_a &= C_{a1} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}, \\
D &= \begin{bmatrix} 0.03 & 0 & 0 & 0.01 \\ 0 & 0.03 & 0 & 0.01 \\ 0 & 0 & 0.03 & 0 \\ 0.01 & 0.01 & 0 & 0.01 \end{bmatrix}, \\
E &= \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}.
\end{aligned} \tag{63}
$$

Because the eigenvalue of the system matrix is $\lambda_1 = \lambda_2 = 0, \lambda_3 = -0.0043 - 0.7843i$, and $\lambda_4 = -0.0043 + 0.7843i$, the system is unstable without a controller.

In the following, this paper jointly designs the event trigger parameter (δ_y, W_y) and (δ_u, W_u) and the control parameter form of the event trigger state feedback, and the closed-loop system (33) is exponentially stable in periodic DoS interference attack.

Other parameter settings are as

$$
\begin{aligned}
\text{miu1} &= \text{miu2} = 1.3, \\
\alpha_1 &= 0.2, \\
\alpha_2 &= 0.3, \\
h &= 0.01, \\
\delta_x &= \delta_u = 0.2, \\
T &= 3, \\
T_{\text{off}}^{\min} &= 2.76, \\
gg &= 50, \\
\gamma_n &= 3\,(n = 1, 2, 3, 4, 5_u).
\end{aligned} \tag{64}
$$

By using Matlab, the parameters of the two-terminal elastic event trigger mechanism (10), (12), (13), and (15) and the gain matrix of the controller (53) can be obtained as follows:

$$
\begin{aligned}
\overline{K} &= \begin{bmatrix} -2.3851 & 1.0982 & -5.1571 & -4.5684 \end{bmatrix}, \\
W_u &= 0.5605, \\
W_y &= \begin{bmatrix} 0.2083 & * & * & * \\ 0.1025 & 0.0814 & * & * \\ -0.0406 & -0.0638 & 5.6805 & * \\ -0.0463 & -0.0447 & 0.1561 & 0.0239 \end{bmatrix}, \\
\overline{B}_c &= \begin{bmatrix} -0.4328 \\ 0.1943 \\ 1.4566 \\ -0.4108 \end{bmatrix}, \\
\overline{A}_c &= \begin{bmatrix} 0.8126 & * & * & * \\ -0.2207 & -0.1238 & * & * \\ 0.3849 & -0.1134 & 3.0924 & * \\ 1.4208 & 0.5436 & 1.0843 & 0.0191 \end{bmatrix}, \\
\overline{D}_c &= \begin{bmatrix} 0.4418 & * & * & * \\ 0.3056 & -1.0061 & * & * \\ -0.4158 & -0.1509 & -3.9104 & * \\ 1.1845 & 0.4818 & 0.0045 & 0.0548 \end{bmatrix}.
\end{aligned} \tag{65}
$$

The internal initial condition system is given $x_0 = \begin{bmatrix} 0.1 & -0.1 & -0.01 & -0.04 \end{bmatrix}^T$, and the simulation time is assumed to be 40*s*. The state graph of DoS attack NCS based on output feedback is shown in Figure 3. Obviously, the system can have good stability through Theorem 3.

Figures 4 and 5 show the data trigger status of the controller and sensor side under the double-ended elastic event trigger mechanism designed in this chapter. The simulation results show that 120 data on the sensor side can meet the conditions of the elastic mechanism (10) and (12), and a total of 132 sampled data on the controller side meet the conditions of the elastic trigger mechanism (13) and (15).

A total of 300 data are required to be sent to the network, and the trigger interval must be 0.1 s. Therefore, after using the method of output feedback, the sensor has 134 data that can meet the conditions of the event mechanism.

As can be seen from Tables 1 and 2, compared with the two-terminal elastic event mechanism and periodic trigger mechanism proposed in this section, it is found that the controller and the sensor side save 55.60% and 60.00% of resources, respectively, and the transmission cycle increases by 0.1273 s and 0.1400 s, respectively. Compared with the single-ended mechanism DoS attack in [28], the controller and sensor side save 4.67% and 55.60% resources, respectively, and the cycle changes are 0.0261 s and 0.1273 s, respectively. Therefore, the double-ended elastic event mechanism proposed in this section meets the performance requirements of the system very well, counteracts the known periodic DoS attacks, saves limited network resources, and improves the antijamming ability of the system.

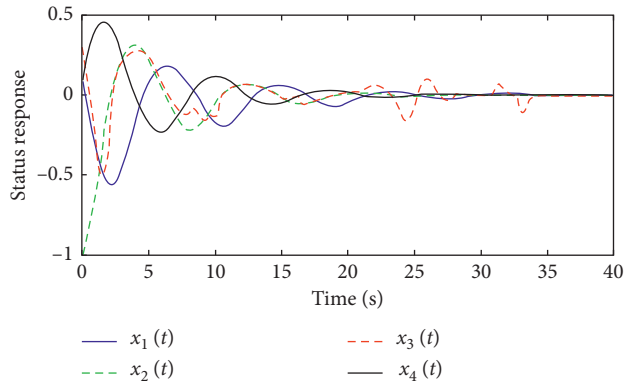


Figure 3: $T = 3$ and $T_{\text{off}}^{\min} = 2.76$ state response curve under output feedback.

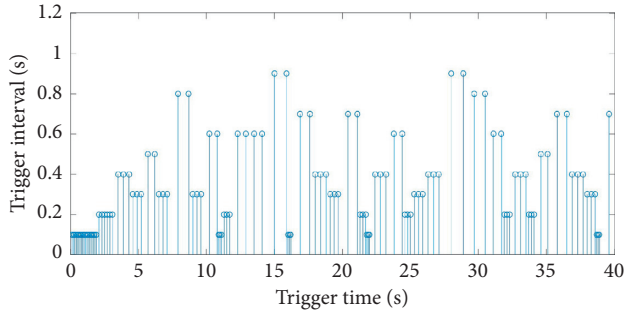


Figure 4: Trigger diagram of sensor side of $T = 3$ and $T_{\text{off}}^{\min} = 2.76$ under elastic trigger mechanism.

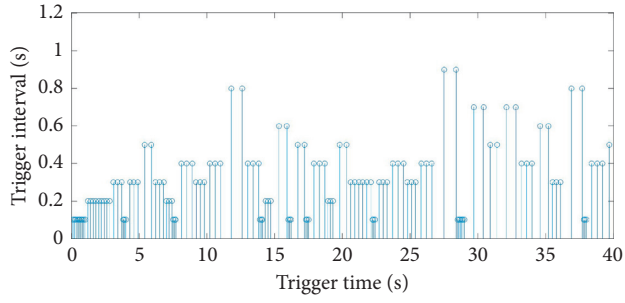


Figure 5: Controller-side trigger diagram of $T = 3$ and $T_{\text{off}}^{\min} = 2.76$ under elastic trigger mechanism under output feedback.

Table 1: Comparison diagram of output feedback data transmission rate under DoS attack based on event mechanism.

Event trigger mechanism	Sensor side (%)	Controller side (%)
Periodic trigger mechanism	100	100
Single-ended event mechanism DoS attack in literature [28]	44.67	—
The double-ended elastic mechanism DoS attack proposed in this chapter	0.00	44.40

In order to show the influence of the interference period T, this section also solves the following optimization problems for the time interval of different T values [0, 40]:

$$\Gamma_{\text{off}}^{\min} = \min\left\{T_{\text{off}}^{\min} | \text{satisfy}\,(4-40)\right\}. \tag{66}$$

In order to verify the influence of event trigger parameters δ_x and δ_u on system stability, Tables 3 and 4 are obtained.

To sum up, it can be concluded that (1) the system is stable; (2) the event trigger mechanism can reduce the traffic

Table 2: Comparison diagram of sending cycle of output feedback data under DoS attack based on event mechanism.

Event trigger mechanism	Sensor side (s)	Controller side (s)
Periodic trigger mechanism	1/10	1/10
Single-ended event mechanism DoS attack in literature [28]	0.2239	1/10
The double-ended elastic mechanism DoS attack proposed in this chapter	0.2500	0.2273

Table 3: λ and ϖ come from different $T_{\text{off}}^{\min}$.

$T_{\text{off}}^{\min}$	4.50	4.56	4.61	4.82	4.97	5.12
λ	0.3653	0.4253	0.4753	0.6853	0.8353	0.9853
ϖ	0.0304	0.0354	0.0396	0.0571	0.0696	0.0821

Table 4: $h_{\max}$ and $T_{\text{off}}^{\min}$ come from different δ_x and δ_u.

$\delta_x = \delta_u$	0.02	0.04	0.06	0.08	0.10	0.12
$h_{\max}$	0.06836	0.06829	0.06825	0.06821	0.06816	0.06809
$T_{\text{off}}^{\min}$	2.76877	2.76883	2.76889	2.76894	2.76900	2.76905

in the system; (3) the event-based output feedback controller does counteract the impact of periodic interference attacks.

7. Conclusion

Because the information of many objects cannot be directly monitored and there are DoS attacks in the network, this section studies the NCS under the output feedback of DoS attacks triggered by double-ended elastic events. First of all, this work proposes an elastic event trigger mechanism transmission scheme that depends on the information of the object and the controller, in order to reduce the burden of computing and communication and, at the same time, counteract the DoS interference attack imposed by the power-limited pulse width modulation (PWM) jammer. Secondly, based on the elastic event trigger mechanism and the DoS attack model, the closed-loop time-delay switching model of the system is established. Then, through LMI technology and Lyapunov knowledge, the stability criterion of H_∞ is obtained, which contains the relationship among elastic event trigger mechanism, DoS attack, stability, and delay, and the sufficient conditions of dynamic output feedback controller and elastic event mechanism are obtained. Finally, through a numerical example, it is proved that the double-ended elastic event trigger mechanism designed in this work can not only counteract the impact of DoS attack interference and save network resources but also can better ensure the performance of the system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Science and Technology Development Program of Jilin Province under Grant no. 20180201090GX.

References

[1] B. Chen, D. W. C. Ho, G. Hu, and L. Yu, "Secure fusion estimation for bandwidth constrained cyber-physical systems under replay attacks," *IEEE Transactions on Cybernetics*, vol. 48, no. 6, pp. 1862–1876, 2018.

[2] L. Zha, E. Tian, X. Xie, Z. Gu, and J. Cao, "Decentralized event-triggered H_∞ control for neural networks subject to cyber-attacks," *Information Sciences*, vol. 457-458, no. 18, pp. 141–155, 2018.

[3] J. Liu, E. Tian, X.-P. Xie, and H. Lin, "Distributed event-triggered control for networked control systems with stochastic cyber-attacks," *Journal of the Franklin Institute*, vol. 1, no. 5, pp. 48–56, 2018.

[4] J. Liu, L. Wei, X.-P. Xie, and D. Yue, "Distributed event-triggered state estimators design for networked sensor systems with deception attacks," *IET Control Theory and Applications*, vol. 13, no. 9, 2014.

[5] D. Ding, Z. Wang, Q.-L. Han, and G. Wei, "Security control for discrete-time stochastic nonlinear systems subject to deception attacks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 5, pp. 779–789, 2018.

[6] H. S. Foroush and S. Martinez, "On triggering control of single-input linear systems under pulse-width modulated dos signals," *SIAMJ Control Optimistic*, vol. 54, no. 3, pp. 3084–3105, 2016.

[7] A. Y. Lu and G. H. Yang, "Input-to-state stabilizing control for cyber-physical systems with multiple transmission channels under denial-of-service," *IEEE Transactions on Automatic Control*, vol. 63, no. 8, pp. 1813–1820, 2018.

[8] H. Sun, C. Peng, T. Yang, H. Zhang, and W. He, "Resilient control of networked control systems with stochastic denial of service attacks," *Neurocomputing*, vol. 270, no. 4, pp. 170–177, 2017.

[9] L. Zha, J.-A. Fang, X. Li, and J. Liu, "Event-triggered output feedback H_∞ control for networked Markovian jump systems

with quantizations," *Nonlinear Analysis: Hybrid Systems*, vol. 24, no. 6, pp. 146–158, 2017.

[10] C. D. Persis and P. Tesi, "Input-to-state stabilizing control under denial-of-service," *IEEE Transactions on Automatic Control*, vol. 60, no. 34, pp. 2930–2944, 2015.

[11] V. S. Dolk, P. Tesi, C. De Persis, and W. P. M. H. Heemels, "Event-triggered control systems under denial-of-service attacks," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 93–105, 2017.

[12] S. Feng, P. Tesi, and C. D. Persis, "Towards stabilization of distributed systems under denial-of-service," in *Proceedings of the 56th Annual Conference on Decision and Control*, pp. 5360–5365, Las Vegas, NV, USA, December 2016.

[13] X. Wang and M. Lemmon, "Event design in event-triggered feedback control systems," in *Proceedings of the 47th IEEE Conference on Decision and Control*, pp. 2105–2110, Cancun, Mexico, December 2008.

[14] A. Eqtami, D. Dimarogonas, and K. Kyriakopoulos, "Event-triggered control for discrete-time systems," in *Proceedings of the 2010 American Control Conference*, pp. 4719–4724, Baltimore, MA, USA, July 2010.

[15] R. Postoyan, A. Anta, D. Nesic, and P. Tabuada, "A unifying lyapunov-based framework for the event-triggered control of nonlinear systems," 2011, http://arxiv.org/abs/1108.5504.

[16] C. De Persis, R. Sailer, and F. Wirth, "On a small-gain approach to distributed event-triggered control," 2010, http://arxiv.org/abs/1010.6148.

[17] C. De Persis and P. Tesi, "Networked control of nonlinear systems under Denial-of-Service," *Systems & Control Letters*, vol. 96, no. 7, pp. 124–131, 2016.

[18] J. Su, R. Xu, S. Yu et al., "Redundant rule detection for software-defined networking," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2735–2751, 2020.

[19] W. Tian, X. Ji, W. Liu et al., "Defense strategies against network attacks in cyber-physical systems with analysis cost constraint based on honeypot game model," *Computers, Materials & Continua*, vol. 60, no. 1, pp. 193–211, 2019.

[20] Y. Xu, T. Xu, and X. Xu, "Research on detection method of interest flooding attack on content centric network," *Computers, Materials & Continua*, vol. 64, no. 2, pp. 1075–1089, 2020.

[21] J. Cheng, J. Li, X. Tang, V. S. Sheng, C. Zhang, and M. Li, "A novel DDOS attack detection method using optimized generalized multiple kernel learning," *Computers, Materials & Continua*, vol. 62, no. 3, pp. 1423–1443, 2020.

[22] H. Chen and S. Kuo, "Active detecting DDOS attack approach based on entropy measurement for the next generation instant messaging app on smartphones," *Intelligent Automation & Soft Computing*, vol. 25, no. 1, pp. 217–228, 2019.

[23] X.-M. Zhang and Q.-L. Han, "Event-based H_∞ filtering for sampled-data systems," *Automatica*, vol. 51, no. 26, pp. 55–69, 2015.

[24] C. Liu and F. Hao, "Dynamic output-feedback control for linear systems by using event-triggered quantisation," *IET Control Theory & Applications*, vol. 9, no. 8, pp. 1254–1263, 2015.

[25] L. Yu, *Robust Control-Linear Matrix Inequality*, Tsinghua University Press, Beijing, China, 2002.

[26] B. L. Zhang, Q. L. Han, X. M. Zhang et al., "Sliding mode control with mixed current and delayed states for offshore steel jacket platform," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 1769–1783, 2014.

[27] B.-L. Zhang, Q.-L. Han, and X.-M. Zhang, "Recent advances in vibration control of offshore platforms," *Nonlinear Dynamics*, vol. 89, no. 2, pp. 755–771, 2017.

[28] X. Chen, Y. Wang, and S. Hu, "Event-based robust stabilization of uncertain networked control systems under quantization and denial-of-service attacks," *Information Sciences*, vol. 459, no. 59, pp. 369–386, 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6684179, 13 pages
https://doi.org/10.1155/2021/6684179

Research Article

PLDP: Personalized Local Differential Privacy for Multidimensional Data Aggregation

Zixuan Shen, Zhihua Xia, and Peipeng Yu

School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China

Correspondence should be addressed to Zhihua Xia; xia_zhihua@163.com

Received 23 November 2020; Revised 24 December 2020; Accepted 2 January 2021; Published 28 January 2021

Academic Editor: Liguo Zhang

The collection of multidimensional crowdsourced data has caused a public concern because of the privacy issues. To address it, local differential privacy (LDP) is proposed to protect the crowdsourced data without much loss of usage, which is popularly used in practice. However, the existing LDP protocols ignore users' personal privacy requirements in spite of offering good utility for multidimensional crowdsourced data. In this paper, we consider the personality of data owners in protection and utilization of their multidimensional data by introducing the notion of personalized LDP (PLDP). Specifically, we design personalized multiple optimized unary encoding (PMOUE) to perturb data owners' data, which satisfies ϵ_{total}-PLDP. Then, the aggregation algorithm for frequency estimation on multidimensional data under PLDP is developed, which is described in two situations. Experiments are conducted on four real datasets, and the results show that the proposed aggregation algorithm yields high utility. Moreover, case studies with four real datasets demonstrate the efficiency and superiority of the proposed scheme.

1. Introduction

In big data era, companies and institutions have noticed the big value of the data and are highly motivated to collect high-dimensional crowdsourced data to make data-driven decisions. The collection and analysis of data are beneficial to companies as well as the society; however, the data owners' privacy makes the biggest concern. In recent years, local differential privacy (LDP) [1, 2] has been found practical value in collection and utilization of data owners' data with the privacy preserved. In an LDP scheme, the data owners perturb their sensitive data before data outsourcing and then report the perturbed data to the server. In this way, the server cannot infer the owners' actual data with strong confidence, however, can still make the accurate estimation of data distribution as it was inferred from the unperturbed data. Considering the desirable properties of LDP, it has been adopted in practice and performs excellently. For example, Apple Inc. collects users' emoji records to discover the popular emojis [3] under LDP. Microsoft also designs the LDP scheme to collect application telemetry to improve user experience [4].

Although the existing LDP schemes are good solutions for data distribution estimation with privacy protected, they ignore data owners' personal privacy requirements. Specifically, in most existing LDP schemes [5, 6], all the owners perturb their different dimensions of data with the same privacy budget, which is set by the server, while it is the truth that different data have a different importance to each owner. In the real world, data owners must have different privacy requirements for their data. Furthermore, if the privacy protection level provided by the server is lower than an owner's need, the owner may be reluctant to share the data. Therefore, it is worth designing the LDP scheme with the personalized privacy allocation mechanism.

In this paper, we propose a multidimensional joint distribution estimation scheme with the personalized local differential privacy (PLDP), in which each data owner has his personal privacy requirement for each dimension of data. Specifically, the server sets an average privacy budget $\varepsilon_{\text{average}}$ to all of its data owners $O = \{o_1, o_2, \ldots, o_N\}$, and the owner o_i can split and allocate $m_i \times \varepsilon_{\text{average}}$ to each dimension of his m_i-dimensional data personally. Then, the data of different dimensions of the owner o_i will be perturbed with the

different privacy budgets. The data owners need not to report their privacy allocation to the server, i.e., the server only holds data owners' perturbed data and their total privacy budgets. Thus, a PLDP scheme provides stronger privacy assurance than the normal LDP scheme.

Moreover, a single data owner may not have the data of all the dimensions required by the server. In our scheme, an owner is allowed to report the data in a part of the dimensions. But, in this way, it is likely that the record of some high-dimensional data could not exist. Then, it is difficult for the server to estimate such dimensions of data. To address it, an aggregation algorithm is developed for joint distribution estimation on multidimensional data under PLDP. The contributions can be summarized as follows:

(1) We propose a new privacy notion called personalized local differential privacy (PLDP), which allows personalized privacy protection for different inputs than LDP. It has higher security and is more personalized than traditional LDP.

(2) We design a perturbed mechanism personalized multiple optimized unary encoding (PMOUE) and develop the aggregation algorithm for frequency estimation on multidimensional data under PLDP to estimate the joint distribution of multidimensional data.

(3) We propose ε_w to measure the personalized privacy protection level of multidimensional data for a single data owner o_i.

(4) Experiments on four real datasets validate the efficiency and superiority of PLDP. Compared with LoPub, PLDP outperforms it with high security, high privacy protection, considerable data utility, and low time consumption.

1.1. Roadmap. In Section 2, we review previous work related to LDP. Then, we give the preliminaries of LDP in Section 3. In Section 4, we describe the problem statement, define the new notion PLDP, and compare it to the traditional LDP. Then, we design the perturbation mechanism PMOUE and develop the aggregation algorithm to estimate the joint distribution of multidimensional data under PLDP. Experimental results are shown in Section 5. Finally, we present the conclusions of this paper in Section 6.

2. Related Work

Differential privacy is a rigorous mathematical definition of privacy for securely sharing the statistic of a dataset on a server [7]. When a requester requests a statistic value of a dataset, the server will calculate the value and then send a disturbed but usable value to the requester. An algorithm is said to be differentially private if the requester cannot infer any single data in the dataset by analyzing the statistic values. In a DP scheme, the server is supposed to be trusted and has all data owners' raw data. However, in the big data era, the data on the server are collected from users. The collection of data provides much useful information to the public; however, the data owners' privacy becomes the big concern as the server generally cannot be fully trusted in the real world. Accordingly, the local differential privacy (LDP) schemes are proposed, in which the owners perturb their data locally and then send the perturbed version to the server. With the perturbed data from data owners, the server can still estimate the distribution of the data.

As the first application of LDP to a real-world problem, Erlingsson et al. [8] proposed RAPPOR to securely estimate the character frequency in a set of strings. Specifically, each data owner uses k hash functions to hash his string onto k Bloom filters [9]. Then, two randomized response (RR) mechanisms, i.e., permanent randomized response and instantaneous randomized response, are proposed to perturb the bits in k Bloom filters. After receiving the perturbed data from the data owners, the server combines the mapping matrix with the LASSO regression [10] to estimate the character frequency. Kim et al. [11] also used randomized response mechanism to collect indoor position records for estimating the density of the specified indoor area. Several fixed points are selected in the indoor area, and each position is represented by its nearest point and denoted as a binary string. Then, the binary string is perturbed by the RR mechanism and then reported. Some researchers tried to reduce the communication cost by reporting a randomly selected bit from the binary string. The bit is also perturbed by RR mechanism. Then, the maximum likelihood estimation [12] and LASSO regression [13] are utilized to the data frequency.

The schemes mentioned above are designed for one-dimensional data. Considering the multidimensional data generally contains more valuable information, several LDP schemes for the multidimensional data are proposed recently. Ren et al. proposed an LDP scheme, named LoPub [14], to synthesize the high-dimensional dataset with the similar distribution to the real dataset. The data are encoded by Bloom filters and perturbed by RR mechanism as in [8]. In order to decrease the computation complexity, the authors tried to calculate the joint distribution of the small set of attributes at first and then calculate the joint distribution of the all attributes by multiplication. Specifically, the attributes with high mutual correlation are clustered together, generating the attribute clusters. The attribute clusters are considered to be independent of each other. Then, the joint attribute distribution within the clusters is calculated by the expectation-maximization algorithm [15] and LASSO regression. The joint distribution of all attributes is obtained by multiplying the distribution of all clusters. Zhang et al. proposed an LDP scheme, named CALM [16], to estimate marginal tables of multidimensional data. Instead of directly generating all marginal tables, the authors constructed many subsets of attributes and chose the randomization algorithm according to the marginal sizes. Expectation-maximization algorithm is used to estimate the marginal distribution. Since some attributes should exist in different subsets, the authors considered the marginal distributions to provide more accurate estimation.

Some researchers considered that different data would have different privacy requirements. Gu et al. proposed an

input-discriminative LDP scheme, in which the server sets different privacy budgets to attribute values. An input-discriminative unary encoding is designed to perturb the attribute value with the specified budgets. In [17, 18], the server provides several privacy budgets and owner can choose one budget for himself to perturb his data. Then, both the perturbed data and the selected budget are reported to the server. The server will estimate the distributions by grouping the data according to the budget. In [19], it is the data owner who decides the privacy budget for his own data; however, it still needs to report the privacy budget to the server for frequency estimation. The schemes [17–19] have tried to consider the personal privacy requirement, but all of them are designed for one-dimensional data. In addition, the server needs to know the privacy budgets that are applied to the disturbed data to estimate the data distribution, which would also expose privacy to the server.

To sum up, there is still a gap in the research on the LDP schemes that support the personalized privacy requirement for multidimensional data. Our goal is to design a scheme where the data owners can choose and perturb their data personally and the server can make a good distribution estimation with such perturbed data.

3. Preliminaries

3.1. Differential Privacy. Differential privacy (DP) is a rigorous mathematical definition of privacy for securely sharing the statistic of a dataset on a server [7]. In a DP scheme, the data owners report their raw data to the trusted server. Then, the server sends a perturbed query result to the requester by adding noise, such as Laplace noise [20]. In this way, the requester is unable to infer much about any single data in the dataset. A formal definition of DP is presented as follows.

Definition 1. differential privacy (DP) [7]). For a given privacy budget $\varepsilon \in R^+$, a randomized mechanism M satisfies ϵ-DP if and only if any neighboring dataset D, D' differs in at most one record and all subsets Y ⊆ range (M) and it has

$$\frac{\Pr[M(D) \in Y]}{\Pr[M(D') \in Y]} \leq e^{\varepsilon}. \tag{1}$$

3.2. Local Differential Privacy. In DP schemes, the server is fully trusted and can hardly apply to the case of privacy-aware crowdsourced systems. Accordingly, the local differential privacy (LDP) scheme is proposed, in which the data owners perturb their data locally and then send the perturbed version to the server. After receiving the perturbed data from owners, the server can still estimate the distribution of the data without knowing much about the data of the single owner. The formal definition of DP can be described as follows.

Definition 2. local differential privacy (LDP) [2]). For a given privacy budget $\varepsilon \in R^+$, a randomized mechanism M satisfies ε-LDP if and only if for any pair of inputs x, x' and any possible output y∈Range (M), it has

$$\frac{\Pr[M(x) = y]}{\Pr[M(x') = y]} \leq e^{\varepsilon}. \tag{2}$$

Theorem 1. *Sequential composition of LDP [21]). If the randomized mechanism M_i satisfies ε_i-LDP, for the given privacy budgets $\varepsilon_i \in R_i^+$, $i = 1, 2, \cdots, k$. Then, their sequential combination $M = (M_1, M_2, \ldots, M_k)$ satisfies $(\sum_{i=1}^{k} \varepsilon_i)$-LDP.*

Proof. For any inputs x, x', and output y, we have

$$\begin{aligned}\Pr[M(x) = y] &= \Pr[M_1(x_1) = y_1]\Pr[M_2(x_2) = y_2], \ldots, \Pr[M_k(x_k) = y_k] \\ &\leq e^{\varepsilon_1}\Pr[M_1(x_1') = y_1]e^{\varepsilon_2}\Pr[M_2(x_2') = y_2], \ldots, e^{\varepsilon_k}\Pr[M_k(x_k') = y_k] \\ &= e^{\sum_{i=1}^{k}\varepsilon_i}\Pr[M(x') = y].\end{aligned} \tag{3}$$

According to Theorem 1, a given total privacy budget $\varepsilon_{\text{total}}$ can be split into ε_i with $\varepsilon_{\text{total}} = \sum_{i=1}^{k} \varepsilon_i$, where each ϵ_i denotes the privacy budget of a randomized mechanism M_i. □

3.3. Optimized Unary Encoding

3.3.1. Randomized Response. Randomized response (RR) [22, 23] is a mainstream perturbation mechanism for LDP. The main idea is to give a random answer to a sensitive question. Accordingly, RR will disturb an input x to an output y as

$$P(y \mid x) = \begin{cases} x, & \text{w.p. } p, \\ x', & \text{w.p. } 1 - p. \end{cases} \tag{4}$$

Specifically, the interviewee gives the genuine answer x with probability p and gives the opposite answer x' with probability 1-p. In this way, RR satisfies $\ln(p/(1-p))$-LDP.

Considering RR only works for binary data, some perturbation mechanisms generalize and optimize it, such as direct encoding (DE) [24], histogram encoding (HE) [25], unary encoding (UE) [8, 26], and local hashing (LH) [6, 27].

In UE, an input x is encoded as a length-l binary vector, with only the bit corresponding to x set to 1. Then, the binary vector will be perturbed bit by bit with probability p and q as follows:

$$\Pr(\mathrm{UE}(x[i]) = 1) = \begin{cases} x[i] = 1, & \text{w.p. } p, \\ x[i] = 0, & \text{w.p. } q. \end{cases} \tag{5}$$

If UE uses $p + q = 1$, it becomes symmetric unary encoding (SUE) [8]. If UE uses optimized choices of p and q, it becomes optimized unary encoding (OUE) [26].

3.3.2. Optimized Unary Encoding. Optimized unary encoding (OUE) [26] converts the input $x = i$ into a binary vector $v = (0, \ldots, 0, 1, 0 \ldots, 0)$ with length-l, where the i-th bit is 1. Then, v is perturbed as follows:

$$\begin{cases} \Pr(y[i] = 1 \mid v[i] = 1) = p = \dfrac{1}{2}, \\ \Pr(y[i] = 0 \mid v[i] = 1) = 1 - p = \dfrac{1}{2}, \\ \Pr(y[i] = 1 \mid v[i] = 0) = q = \dfrac{1}{e^{\varepsilon} + 1}, \\ \Pr(y[i] = 0 \mid v[i] = 0) = 1 - q = \dfrac{e^{\varepsilon}}{e^{\varepsilon} + 1}. \end{cases} \tag{6}$$

In this way, OUE satisfies ϵ-LDP.

Proof. For any pair of inputs x and x', we denote their corresponding vector as v and v', respectively, and denote the output vector as y. Then, we have $\theta = ((\Pr(y \mid v))/(\Pr(y \mid v'))) = \prod_{i=1}^{l} ((\Pr(y[i] \mid v[i]))/(\Pr(y[i] \mid v'[i])))$. According to the OUE, there is only one '1' bit in both the v and v'. Let us assume that $v[s] = v'[t] = 1$, and there are four different cases for θ since the vector y could have four different possible values, which is listed as follows:

$$\theta = \begin{cases} \theta_1 = \dfrac{\Pr(y[s] = 0 \mid v[s] = 1)\Pr(y[t] = 0 \mid v[t] = 0)}{\Pr(y[s] = 0 \mid v'[s] = 0)\Pr(y[t] = 0 \mid v'[t] = 1)}, & \text{if } y[s] = 0, y[t] = 0, \\ \theta_2 = \dfrac{\Pr(y[s] = 0 \mid v[s] = 1)\Pr(y[t] = 1 \mid v[t] = 0)}{\Pr(y[s] = 0 \mid v'[s] = 0)\Pr(y[t] = 1 \mid v'[t] = 1)}, & \text{if } y[s] = 0, y[t] = 1, \\ \theta_3 = \dfrac{\Pr(y[s] = 1 \mid v[s] = 1)\Pr(y[t] = 0 \mid v[t] = 0)}{\Pr(y[s] = 1 \mid v'[s] = 0)\Pr(y[t] = 0 \mid v'[t] = 1)}, & \text{if } y[s] = 1, y[t] = 0, \\ \theta_4 = \dfrac{\Pr(y[s] = 1 \mid v[s] = 1 \mid)\Pr(y[t] = 1 \mid v[t] = 0)}{\Pr(y[s] = 1 \mid v'[s] = 0)\Pr(y[t] = 1 \mid v'[t] = 1)}, & \text{if } y[s] = 1, y[t] = 1. \end{cases} \tag{7}$$

Then, according to Formula (6), we have

$$\theta = \begin{cases} \theta_1 = \dfrac{(1-p)(1-q)}{(1-q)(1-p)} = 1, \\ \theta_2 = \dfrac{(1-p)q}{(1-q)p} = \dfrac{q}{1-q}, \\ \theta_3 = \dfrac{p(1-q)}{q(1-p)} = \dfrac{1-q}{q}, \\ \theta_4 = \dfrac{pq}{qp} = 1, \end{cases} \tag{8}$$

and $q = (1/(e^{\varepsilon} + 1))$. Since ϵ is definitely a positive number, we have $q < (1/2)$. Then, it is derived that

$$\theta \le \theta_3 = \frac{1-q}{q} = e^{\varepsilon}, \tag{9}$$

which demonstrates that OUE satisfies ϵ-LDP. □

3.4. Least Absolute Shrinkage and Selection Operator Regression. For real multidimensional data, its joint distribution may be sparse. To estimate the joint distribution of real multidimensional data, the least absolute shrinkage and selection operator (LASSO) regression is usually used. Since when solving a multivariate objective function as follows,

$$\widehat{\beta} = \min_{\beta} \|y - X\beta\|_2^2, \tag{10}$$

where y is the label, X is the vector of an input sample, and $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ is the vector of regression coefficients; the overfitting problem may occur. To address it, LASSO regression adds L1-norm as a penalty term after the objective function. Then, constructing a penalty function as follows,

$$\beta^* = \min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1, \tag{11}$$

where λ is a data dependent parameter.

In this way, when updating the regression coefficient β_i to β_i^* with learning rate η, $\beta_i^* = \beta_i - \eta((\partial(\beta^*))/(\partial(\beta_i)))$. Since $(\partial(\lambda\|\beta\|_1)/\partial(\beta_i)) = \lambda$ or $-\lambda$, the updated β_i^* is likely to be zero. In this way, LASSO regression forces the sum of the absolute value of the regression coefficients to be less than a fixed value, which keeps certain coefficients to be set to zero and accordingly results in a simpler model that does not include those zero coefficients. As a result, LASSO regression is quite suitable to solve the sparse linear regression because it can compress the coefficients of variables and make some regression coefficients become zero.

4. PLDP: Personalized Local Differential Privacy

4.1. System and Threat Model. In the existing multidimensional LDP schemes, the server sets a fixed privacy budget to each attribute to facilitate the estimation of the joint distribution. Since the same attributes of all owners are assigned the same amount of privacy budget, it ignores the personal privacy requirement of owners. However, in real life, the privacy levels of attributes could be distinct to the data owners, i.e., each owner has his personal privacy requirements for his data record. Therefore, it could be a better way to assign different budgets, which are personally decided by the data owner, to each attribute. In this paper, we proposed a multidimensional LDP scheme where the data owner can assign the personal privacy budget to the data he plans to upload according to his personal privacy requirement.

4.1.1. System Model. This paper considers a server that would like to collect a set of data records with d attributes from various data owners. The attributes are denoted as $\{A_1, A_2, \ldots, A_d\}$. To preserve the privacy, the owners are allowed to perturb the data according to his personal privacy requirement before uploading the data. Our goal is to enable the server to estimate the joint distribution of attributes from the perturbed data to benefit the public. Specifically, our system model involves a server and a set of data owners $O = \{o_1, o_2, \ldots, o_N\}$, where the owner o_i submits a perturbed data record $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ to the server. In practice, a data owner does not have to possess or be willing to upload the all of d-dimensions of data. Thus, our scheme allows the owner to upload his record with some dimensions of data empty. After receiving the data records, the server puts the owners that report the same dimensions of data into the same group. Then, the joint distribution of data will be conducted within the group or by aggregating several groups. The system model of the proposed scheme can be illustrated in Figure 1.

4.1.2. Threat Model. We assume that the server is untrusted, since the server may leak the data owners' privacy, such as being hacked, or selling owners' data to a third party for profit. In this way, the adversary is assumed to possess a perturbed database with d attributes collected by the server, the principle of the perturbation mechanism, and the total privacy budgets of all owners.

4.2. Definition of PLDP and Its Relationships with LDP. In this paper, we propose a new LDP notion, named personalized local differential privacy (PLDP). In a PLDP scheme, the server sets an average privacy budget $\varepsilon_{\text{average}}$ for all of the data owners. If an owner plans to report a data record with m elements unemptied, the owner will hold a total amount of privacy budget $\varepsilon_{\text{total}} = m \times \varepsilon_{\text{average}}$. Then, the owner can personally allocate $\varepsilon_{\text{total}}$ to m elements according to his privacy requirement. In addition, the privacy budget allocation will not be reported to the server, which enhances the security of the users' data.

Definition 3. personalized local differential privacy (PLDP)). For a given average privacy budget $\varepsilon_{\text{average}} \in R^+$, each data owner allocates the total privacy budget $\varepsilon_{\text{total}} = m \times \varepsilon_{\text{average}}$ personally to his data record with m unemptied elements. Denote the m separate privacy budgets as $\varepsilon_i > 0$, $i = 1, \ldots, m$, and it definitely has $\varepsilon_{\text{total}} = \sum \varepsilon_i$. Then, a randomized mechanism M satisfies $\varepsilon_{\text{total}}$-PLDP if and only if for any pair of records x', x'' with the same unemptied elements, and any output $y \in \text{Range}(M)$, we have

$$\frac{\Pr[M(x') = y]}{\Pr[M(x'') = y]} \leq e^{\varepsilon_1' + \varepsilon_2' + \cdots + \varepsilon_m'} = e^{\varepsilon_1'' + \varepsilon_2'' + \cdots + \varepsilon_m''} = e^{\varepsilon_{\text{total}}}, \tag{12}$$

where ε_i' and ε_i'' are the privacy budgets allocated to x' and x'', respectively.

4.2.1. Relationships with LDP. In a multidimensional data scenario, if the privacy budgets for each attribute are the same, i.e., $\varepsilon_1' + \varepsilon_2' + \cdots + \varepsilon_m' = \varepsilon_1'' + \varepsilon_2'' + \cdots + \varepsilon_m''$, the PLDP becomes LDP. It means PLDP is a generalized version of LDP. The relationship between LDP and PLDP can be specified by the following two theorems.

Theorem 2. *For any pair of records x', x'' with m attributes, if a perturbation mechanism M satisfies ε-LDP, it also satisfies ε_{total}-PLDP with $\varepsilon_{total} = \varepsilon$.*

Proof. In a common LDP scheme without considering the personal privacy requirement of data owner, the privacy budget of each attribute equals to (ε/m). It can be considered as having all the ε_i equal to (ε/m) in $\varepsilon_{\text{total}}$-PLDP with $\varepsilon_{\text{total}} = \sum_{i=1}^{m} \varepsilon_i = \varepsilon$.

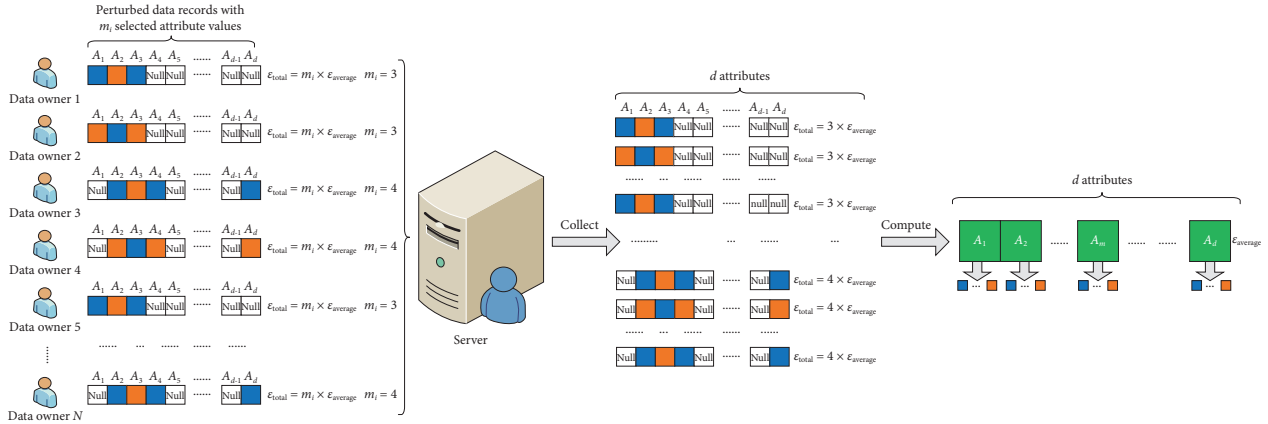


Figure 1: The architecture of the system model.

Theorem 2 indicates that LDP is a special case of PLDP. The PLDP can be considered as a generalization of LDP. □

Theorem 3. *For any pair of records x', x'' with m attributes, if a perturbation mechanism M satisfies ε_{total}-PLDP, it also satisfies ε-LDP with $\varepsilon = m \times min\ (\varepsilon_i) \le \varepsilon_{total}$, $i = 1, \ldots, m$, where ε_i denotes the privacy budget of i-th attribute.*

Proof. In a common LDP scheme for the multidimensional data record, each attribute is put on the same privacy budget. To guarantee the privacy, we should put the smallest privacy budget min (ε_i) to all of the attributes to get an LDP scheme with $\varepsilon = m \times \min\ (\varepsilon_i) \le \varepsilon_{\text{total}}$ under the same perturbation mechanism. □

4.2.2. Security Comparison. In a PLDP scheme, the privacy budget allocation is also a kind of privacy and thus not reported to the server. In addition, in a PLDP scheme, the owner can freely allocate his privacy budget to the attributes. When a small budget is put on an attribute, it means the attribute is very sensitive to the data owner and the privacy budget on this attribute also weighs a lot. Nevertheless, if a large budget is put on an attribute, it means the attributes are not so important and the privacy budget on this attribute may not weigh much. Here, we definite a notion to quantize the amount of weighted privacy budget, which can be regarded as the real privacy budget to a data owner in the PLDP scheme.

Definition 4 (weighted privacy budget ε_w). With a given total privacy budget $\epsilon_{\text{total}} = m \times \varepsilon_{\text{average}}$ from the server, the data owner splits it into m parts denoted as ε_i, $i = 1, \ldots, m$. Then, the weighted privacy budget of the data owner can be calculated as

$$\varepsilon_w = \sum_{i=1}^{m} \varepsilon_i w_i, \tag{13}$$

where $w_i = 1 - ((\varepsilon_i - \varepsilon_{\min})/\varepsilon_{\text{total}})$ and $\epsilon_{min} = \min\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m\}$. It means that an ε_i will be multiplied by a smaller weight if ε_i is larger than $\varepsilon_{\min}$. Generally, the weighted privacy budget is smaller than ϵ_{total}, which indicates a PLDP scheme provides better privacy protection than an LDP scheme. The illustration of personalized privacy budget is shown in Figure 2.

4.3. Perturbation Mechanism for PLDP. In this section, we adapt the OUE to generate a perturbation mechanism for PLDP, which we name as personalized multiple optimized unary encoding (PMOUE), since there are multiple attributes in our scenario and the privacy budget can be personally allocated. We denote a data record with m unemptied attributes as $X = \{x_1, x_2, \ldots, x_m\}$, and the attribute x_i has l_i candidate values. According to OUE, the attribute $x_i = k$ is encoded to be a binary vector $v_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ with the length l_i. The k-th bit in v_i is set to be 1, and the remaining bits are set to be 0. In order to protect the owner's data record, the bits in v_i are randomly flipped according to Algorithm 1, generating the perturbed element y_i. The parameters p and q in formula (6) depend on the privacy budget ϵ_i that are allocated to the element x_i. Finally, the data owner concatenates all of the y_i to be $Y = \{y_1, y_2, \ldots, y_m\}$ and sends Y to the server. The specific perturbation mechanism is described in Algorithm 1.

Theorem 4. *The perturbation mechanism PMOUE satisfies ϵ_{total}-PLDP, where $\epsilon_{total} = m \times \varepsilon_{average}$.*

Proof. For any inputs x', x'' with the same m unemptied attributes denoted by x_i' and x_i'', which are encoded to v_i' and v_i'', and the output y with m valid elements denoted by y_i, $i = 1, \ldots, m$, we have $\theta = ((\Pr(y|x')))/(\Pr(y|x'')) = \prod_{i=1}^{m} ((\Pr(y_i|x_i')))/(\Pr(y_i|x_i'))) = \prod_{i=1}^{m}\prod_{j=1}^{l_i} ((\Pr(y_i[j]|v_i'[j]))/(\Pr(y_i[j]|v_i''[j])))$.

According to formula (9), we have

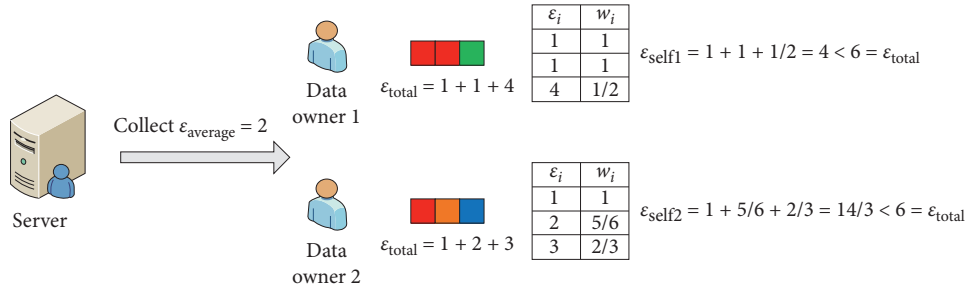


Figure 2: Illustration of personalized privacy budget.

$$\theta \le \prod_{i=1}^{m} \frac{\Pr(y_i[s_i] = 1 \mid v_i'[s_i] = 1)\Pr(y_i[t_i] = 0 \mid v_i'[t_i] = 0)}{\Pr(y_i[s_i] = 1 \mid v_i''[s_i] = 0)\Pr(y_i[t_i] = 0 \mid v_i''[t_i] = 1)}$$

$$= \prod_{i=1}^{m} \frac{p_i'(1 - q_i')}{q_i''(1 - p_i'')}$$

$$= \prod_{i=1}^{m} \frac{1 - q_i'}{q_i''}$$

$$= \prod_{i=1}^{m} \frac{e^{\varepsilon_i'}\left(e^{\varepsilon_i''} + 1\right)}{e^{\varepsilon_i'} + 1}$$

$$= e^{\sum_{i=1}^{m} \varepsilon_i'} \prod_{i=1}^{m} \frac{e^{\varepsilon_i''} + 1}{e^{\varepsilon_i'} + 1}$$

$$= e^{\varepsilon_{total}} \frac{\sum_{i=1}^{m} \sigma_i'' + 1}{\sum_{i=1}^{m} \sigma_i' + 1}, \tag{14}$$

where $\sum_{i=1}^{m} \sigma_i'$ and $\sum_{i=1}^{m} \sigma_i''$ are the m elementary symmetric polynomials about the variables $e^{\varepsilon_1'}, e^{\varepsilon_2'}, \cdots, e^{\varepsilon_m'}$ and $e^{\varepsilon_1''}, e^{\varepsilon_2''}, \cdots, e^{\varepsilon_m''}$ separately, i.e., $\sigma_1' = e^{\varepsilon_1'} + e^{\varepsilon_2'} + \cdots + e^{\varepsilon_m'}$, $\sigma_2' = e^{\varepsilon_1'}e^{\varepsilon_2'} + e^{\varepsilon_1'}e^{\varepsilon_3'} + \cdots + e^{\varepsilon_{m-1}'}e^{\varepsilon_m'}$, $\cdots$, $\sigma_m' = e^{\varepsilon_1'}e^{\varepsilon_2'}, \ldots, e^{\varepsilon_m'}$ [28].

Because $\lim_{x \longrightarrow 0} e^x = x + 1$, so $((\Pr(y \mid x))/(\Pr(y \mid x')))$

$$\le e^{\varepsilon_{total}} \frac{\sum_{i=1}^{m}(\varepsilon_i'' + 1) + \sum_{j=1}^{m}\sum_{i=1, i\neq j}^{m}\left(\varepsilon_i'' + \varepsilon_j'' + 1\right) + \cdots + \sum_{i=1}^{m}\varepsilon_i'' + 1 + 1}{\sum_{i=1}^{m}(\varepsilon_i' + 1) + \sum_{j=1}^{m}\sum_{i=1, i\neq j}^{m}\left(\varepsilon_i' + \varepsilon_j' + 1\right) + \cdots + \sum_{i=1}^{m}\varepsilon_i' + 1 + 1}$$

$$= e^{\varepsilon_{total}} \frac{\varepsilon_{total} + m + (m-1)\varepsilon_{total} + m + \cdots + \varepsilon_{total} + 1 + 1}{\varepsilon_{total} + m + (m-1)\varepsilon_{total} + m + \cdots + \varepsilon_{total} + 1 + 1}$$

$$= e^{\varepsilon_{total}}. \tag{15}$$

□

4.4. Estimation of the Joint Distribution. After receiving $Y = \{y_1, y_2, \ldots, y_m\}$ from each data owner, the server can estimate the joint distribution of multidimensional data. Here, we describe the estimation in two situations: (1) the frequency estimation for k-dimensional data that is included in some records, and (2) the frequency estimation for k-dimensional data that is not included in any records.

4.4.1. Situation 1. Since the k dimensions of data are included in some records, we group these records together for the frequency estimation. As presented in Section 3.3, the k dimensions of data are encoded to be the binary vector with the length $t = \sum_{i=1}^{k} l_i$, where l_i is the number of candidate values of the i-th attribute. Assuming there are s records that have these k dimensions, the frequency will be estimated from a binary matrix denoted as $\widehat{C}_{s\times t}$. Firstly, the server counts the number of '1' in each column, generating a vector $(\widehat{c}_1, \widehat{c}_2, \ldots, \widehat{c}_t)$. Then, to obtain an unbiased estimation, maximum likelihood estimation [29] is used to calibrate $\widehat{c}_i$ as follows:

$$c_i = \frac{\widehat{c}_i - Gq}{p - q} = \frac{\widehat{c}_i - \left(s/\left(e^{\varepsilon_{average}} + 1\right)\right)}{(1/2) - \left(s/\left(e^{\varepsilon_{average}} + 1\right)\right)}, \tag{16}$$

where c_i denotes the occurrence number of each candidate value, indicating the individual distribution of the attributes. G is the total number of records.

With the distribution of each dimension of data, the joint distribution of the k-dimensional data is estimated by LASSO regression. Firstly, the server encodes the candidate values of attributes to be binary string according to OUE. Next, the candidate values of different attributes are connected by Cartesian product and then transposed, resulting in the candidate matrix $M_{t\times r}$, where $t = \sum_{i=1}^{k} l_i$ is the bit length of the candidate value of the k-dimensional data and $r = \prod_{i=1}^{k} l_i$ is the total number of candidate values. Ensemble the candidate value frequencies of the k-dimensional data to be a vector $P = (p_1, p_2, \ldots, p_r)^T$, and we have

$$\mathrm{MP} = C, \tag{17}$$

where $C = (c_1, c_2, \ldots, c_t)^T$. Since the candidate value matrix $M_{t\times r}$ and the vector $C_{t\times 1}$ are both known, it seems that the unknown regression coefficient vector $P_{r\times 1}$ can be derived easily. In fact, the total number of joint candidate values r could be very large, but the occurrence frequencies of some candidate values are very small or even close to 0. That is to say, $P_{r\times 1}$ could be quite sparse.

As presented in Section 3.4, LASSO regression is quite suitable to solve the sparse linear regression. In this paper, the solution of frequency vector P in formula (17) is considered as a sparse linear regression problem where a shrunken vector is preferred. Thus, LASSO is used to obtain the frequency vector P. In Figure 3, we illustrate the whole process to estimate the frequency of the k-dimensional data in Situation 1.

(i) **Input:** $X = \{x_1, x_2, \ldots, x_m\}$: the raw data record with m unemptied attributes
ε_i: the privacy budget that allocated to attribute $x_i\,(i = 1, 2, \ldots, m)$
l_i: the number of the candidate values of each attribute $(i = 1, 2, \ldots, m)$
Output: $Y = \{y_1, y_2, \ldots, y_m\}$: the perturbed data record with m attributes

(1) Encoding:

$$v_i = \text{Encode}(x_i, l_i), i = 1, 2, \ldots, m$$

(2) Perturbation:

$$y_i = \begin{cases} \Pr(y_i[j] = 1 | v_i[j] = 1) = p_i = (1/2), \\ \Pr(y_i[j] = 0 | v_i[j] = 1) = 1 - p_i = (1/2), \\ \Pr(y_i[j] = 1 | v_i[j] = 0) = q_i = (1/(e^{\varepsilon} + 1)), \\ \Pr(y_i[j] = 0 | v_i[j] = 0) = 1 - q_i = (e^{\varepsilon}/(e^{\varepsilon} + 1)), \end{cases} \quad j = 1, 2, \ldots, l_i; i = 1, 2, \ldots, m,$$

(3) return $Y = [y_1, y_2, \ldots, y_m]$.

Algorithm 1: Personalized multiple optimized unary encoding (PMOUE).

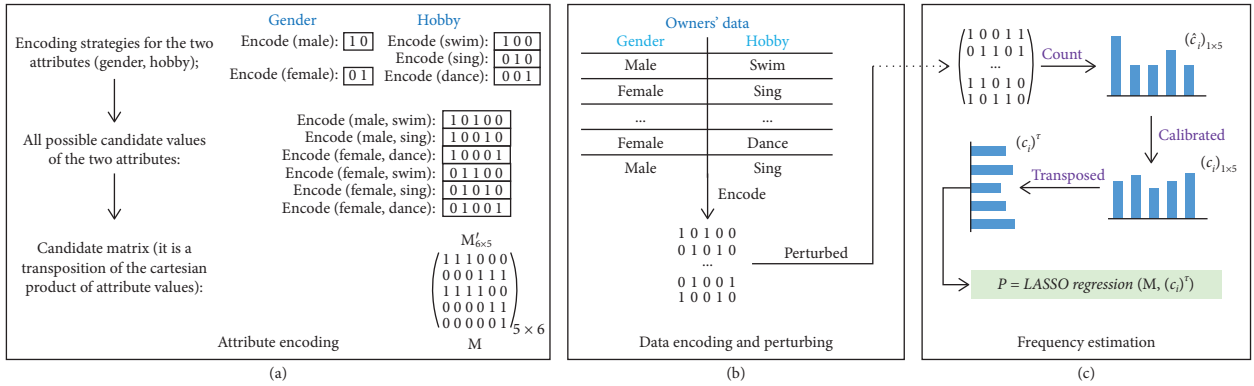


Figure 3: Illustration of LASSO scheme. (a) Attribute encoding. (b) Data encoding and perturbing. (c) Frequency estimation.

As shown in Figure 3, we consider an example with two attributes, i.e. gender and hobby, which, respectively, have two and three candidate values. Firstly, the candidate values are encoded to be the 2-bit and 3-bit binary vectors. The binary vectors of different candidate values are connected by Cartesian product, resulting in a matrix M' which then is transposed to be the candidate matrix M. Next, the data owners encode their data according to the encoding strategy and then perturb and upload the data to the server. In Situation 1, the considered k dimensions of data are included in some records. The server groups these records together to estimate the distribution of the frequency estimation. After receiving the perturbed data, the server counts the '1' bit in each column, resulting in a vector. The server calibrates the vector that indicates the frequencies of attribute values. Finally, the frequency vector P is calculated by LASSO regression algorithm with the M and C.

4.4.2. Situation 2. In this situation, the server wants to estimate the joint distribution of k attributes, denoted as $A = \{a_1, a_2, \ldots, a_k\}$, but there is no record containing all of these k attribute values. Accordingly, we divide the k attributes into two parts, i.e., A_1 and A_2, so that each part of the attributes is contained by some records. The A_1 and A_2 are considered as two multidimensional attributes with the domains Ω_1 and Ω_2, respectively. Then, we can estimate the distributions of A_1 and A_2 by LASSO regression algorithm as in Situation 1 separately and synthesize two distributions together.

The first step is to choose a division strategy and here we prefer the unbalanced one. That is to say, we prefer to let $\mathscr{A}_1$ have the most attributes in $\mathscr{A}$ and let $\mathscr{A}_2$ only have a small part of attributes. The second step is to allocate the attributes into $\mathscr{A}_1$ and $\mathscr{A}_2$. Intuitively, we hope $\mathscr{A}_1$ and $\mathscr{A}_2$ are as independent as possible. Then, the joint distribution of $\mathscr{A}_1$ and $\mathscr{A}_2$ can be calculated by direct multiplication. Information entropy [30] is a usual method to measure the amount of information and can also be used to identify the independency of the variable. Generally, a larger amount of information means larger independence. Based on the points above, we calculate the joint distribution of k attributes as described in Algorithm 2 and show its illustration in Figure 4.

5. Evaluation

In this section, we perform the experiments on four real datasets. The experimental results show the efficiency and superiority of our proposed notion PLDP in the aspect of accuracy and efficiency.

(i) **Input**: $\mathscr{A}$: the set of k attributes for which the sever wants to estimate the joint distribution and $\{X_i\}$ is the setof data records with $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, where x_{ij} could be empty.
Output: HD: the joint distribution of $\mathscr{A}$
(1) **for** $r = 1$ to $(k/2)$ **do**
(2) Initialize a result list, $\text{list}_R = \{\varnothing\}$;
(3) Divided $\mathscr{A}$ into two parts: $\mathscr{A}_1$ and $\mathscr{A}_2$, where $|\mathscr{A}_1| = k - r$ and $|\mathscr{A}_2| = r$, and there are $n = \binom{k}{r}$ different divisions;
(4) **for** $i = 1$ to n **do**
(5) For the i-th division, if there are enough records that have all the attributes in $\mathscr{A}_1$ and $\mathscr{A}_2$, the sever estimatesthe joint distributions of $\mathscr{A}_1$ and $\mathscr{A}_2$ by LASSO regression algorithm, respectively. Denote the domains of candidate values of $\mathscr{A}_1$ and $\mathscr{A}_2$ as Ω_1 andΩ_2, respectively, and then the estimated joint distribution can be denoted as $P' = (p_1', p_2', \ldots, p_{|\Omega_1|}')$ and $P'' = (p_1'', p_2'', \ldots, p_{|\Omega_2|}'')$.
(6) Calculate $H' = \sum_{i=1}^{|\Omega_1|} p_i' \log p_i'$, $H'' = \sum_{i=1}^{|\Omega_2|} p_i'' \log p_i''$, and then the total information entropy in this division $H_i = H' + H''$.
(7) Finally, add the triplet $<P', P'', H_i>$ into list_R;
(8) **end for**
(9) **If** $\text{list}_R \neq \{\varnothing\}$ **Break**;
(10) **end for**
(11) Choose the triplet with the largest H_i and calculate the joint distribution with the corresponding P', P'' by the multiplication principle.

Algorithm 2: Information entropy-based multidimensional joint distribution estimation.

Candidate value	Probability
a1b1	0.25
a1b2	0.25
a2b1	0.25
a2b2	0.25

Candidate value	Probability
c1	0.47
c2	0.53

$H(\text{AB}) + H(\text{C}) = 0.60 + 0.30 = 0.90$

(a)

Candidate value	Probability
a1c1	0.30
a1c2	0.20
a2c1	0.20
a2c2	0.30

Candidate value	Probability
b1	0.70
b2	0.30

$H(\text{AC}) + H(\text{B}) = 0.59 + 0.26 = 0.85$

(b)

Candidate value	Probability
b1c1	0.70
b1c2	0.20
b2c1	0.05
b2c2	0.05

Candidate value	Probability
a1	0.65
a2	0.35

$H(\text{BC}) + H(\text{A}) = 0.38 + 0.28 = 0.64$

(c)

Figure 4: Illustration of information entropy-based scheme. (a) AB and C. (b) AC and B. (c) BC and A.

5.1. Experimental Setup

5.1.1. Environment. All the experiments were performed on a machine with Intel Core i5 CPU 2.50 GHz and 8 GB RAM, using Windows 10 and Python 3.5.2.

5.1.2. Datasets. We performed experiments on the following four real datasets, whose parameters are presented in Table 1. Abalone [31] contains the size of the abalone through physical measurements. Adult [31] is extracted by Barry Becker from the 1994 Census database. It contains personal information, such as "age," "relationship," and "money." Bank marketing [32] is related with direct marketing campaigns, which are based on phone calls, of a Portuguese banking institution. Car evaluation [31] contains several basic parameters of the car, which is derived from a simple hierarchical decision model. In order to simplify the experiments, we bucket the continuous and nonnumerical data into the discretized ones.

5.1.3. Evaluation Metrics. The performance of PLDP is measured by the accuracy of the distribution estimation and the time consumption during the estimation. The time consumption contains CPU time and IO cost. As in much previous work, we used the average variant distance (AVD) [33] to evaluate the estimation accuracy, which is defined as

$$\mathrm{AVD}(P, Q) = \frac{\sum_{\omega \in \Omega} |P(\omega) - Q(\omega)|}{2}, \tag{18}$$

where Ω is the domain of the multidimensional variable, $P(\omega)$ denotes the real joint distribution, and $Q(\omega)$ denotes the estimated distribution.

5.1.4. Setting of Privacy Budget. This paper considers the personalized privacy allocation. Here, we use a random function to assign the total privacy budget $\varepsilon_{\text{total}} = m \times \varepsilon_{\text{average}}$ randomly to the m valid elements for each data owner.

5.2. Comparison with LoPub on Adult Dataset. In LoPub [14], the data owners report their whole data with m attributes in their perturbed version. They perturb each dimension of data with a fixed privacy budget which is predefined by the server. After receiving the perturbed data, the server tried to estimate the joint distribution of k attributes $(1 \le k \le m)$ by using EM-based approach (LoPub-1), Lasso-based approach (LoPub-2), and a hybrid approach (LoPub-3). Please note that EM-based approach (LoPub-1) is not used in the case of $k = 1$ since the distribution of one attribute can be estimated directly by maximum likelihood estimation. In [14], the authors set a fixed privacy budget $\varepsilon = 2$ for each dimension of data, and each owner reports a 4-dimensional perturbed data. When the server estimates the joint distribution of k attributes, the total privacy budget equals to $\varepsilon \times k = 2k$. In our experiments, we test our scheme with three different settings to make a full comparison with LoPub [14].

5.2.1. Setting#1. Each data owner o_i reports an m_i-dimensional data perturbed with the total privacy budget $\varepsilon_{\text{total}} = m_i \times \varepsilon_{\text{average}}$, $\varepsilon_{\text{average}} = 2$, and $1 \le m_i \le 4$. Setting#1 is the normal setting of the proposed scheme, where the privacy budgets allocated on the attributes are uncertain. Accordingly, the joint distribution of $k = 1, 2, 3, 4$ attribute(s) could be estimated in both Situation 1 and Situation 2. The AVDs in Setting#1 are averaged from 10 estimations.

5.2.2. Setting#2. Each owner chooses two dimensions of his data randomly and then perturbed the data with a total privacy budget $\varepsilon_{\text{total}} = 2 \times \varepsilon_{\text{average}} = 2 \times 2 = 4$. Then, the joint distribution of $k = 1, 2, 3, 4$ attribute(s) is estimated. In Setting#2, the privacy budget allocated on the k attributes is equal to that in LoPub. In this way, the estimations of 1 and 2 attribute(s) follow the Situation 1 and the estimations of 3 and 4 attributes follow the Situation 2. The AVDs in Setting#2 are averaged from 10 estimations.

5.2.3. Setting#3. Here, each owner still chooses two dimensions of his data randomly to perturb and upload but sets different $\varepsilon_{\text{average}}$ when a different number of attributes are considered for distribution estimation. Specifically, the $\varepsilon_{\text{average}}$ is set to be k when the joint distribution of k attributes is estimated.

Figure 5(a) shows that our scheme achieves comparable accuracy to LoPub-3 despite that the privacy budgets are personally allocated to the attributes. Figure 5(b) shows that our scheme achieves better efficiency than LoPub, as our method needs not to iteratively scan the data owners' records. To sum up, the proposed scheme can provide personalized privacy allocation for each owner, which means a less weighted privacy budget. Accordingly, compared with LoPub, our scheme has higher security, lower time consumption, and comparable estimation accuracy. Note that, the results with regard to LoPub are directly taken from [14]. The test for LoPub-1 with $k = 4$ is not presented in [14] as it consumes too much time.

5.3. Results on Other Three Datasets. In this section, we test our scheme on four real-word datasets under Setting#1 and Setting#2 that are the same as that in Section 5.2. Figure 6 shows that, besides the dataset Adult, our scheme achieves similar accuracy on other three datasets. Our scheme allowed the data owner to only share a part of his data. Thus, the joint distribution of the high-dimensional data can only be obtained by synthesizing that of the low-dimensional data. We divide the attributes into two parts and decide the division by maximizing the sum of information entropies of the two parts. Finally, we synthesize the joint distribution by multiplication principle with the assumption that the two divided parts are independent of each other. This could decrease the estimation accuracy of our scheme.

Table 2 shows the experimental results of aggregating five-dimensional data. The settings of S#1 and S#2 are similar to the settings of Setting#1 and Setting#2. In S#1, each data owner o_i reports an m_i-dimensional data perturbed with the total privacy budget $\varepsilon_{\text{total}} = m_i \times \varepsilon_{\text{average}}$, $\varepsilon_{\text{average}} = 2$, and $1 \le m_i \le 5$. In S#2, each owner chooses three dimensions of

Table 1: Datasets.

Datasets	Attribute characteristics	#.Records (N)	#.Attributes (d)
Adult	Categorical, integer	48842	15
Abalone	Categorical, integer, real	4177	9
Bank marketing	Real	45211	17
Car evaluation	Categorical	1728	7

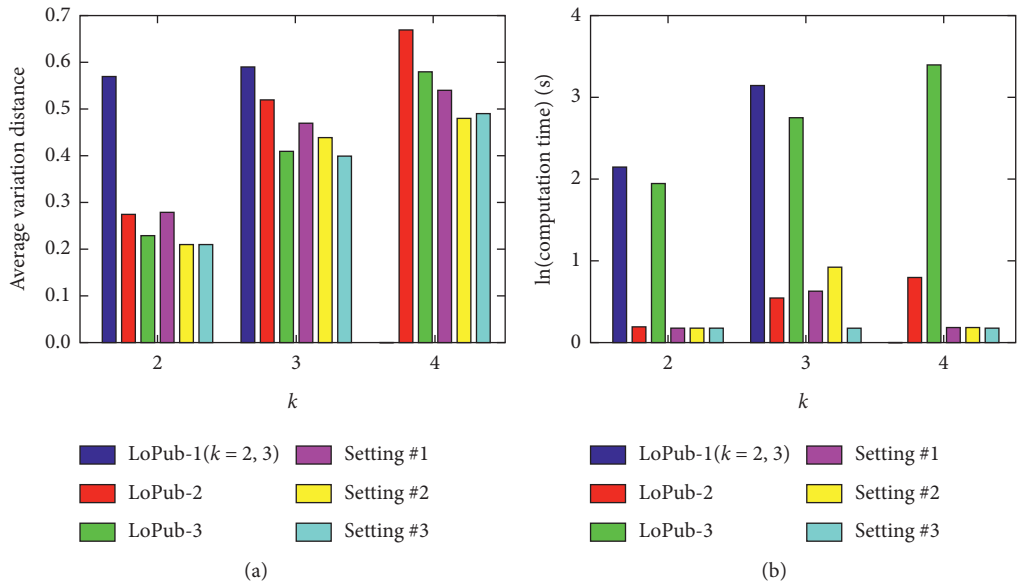


Figure 5: The comparison of the proposed scheme and LoPub on adult dataset. (a) AVD. (b) Computation time.

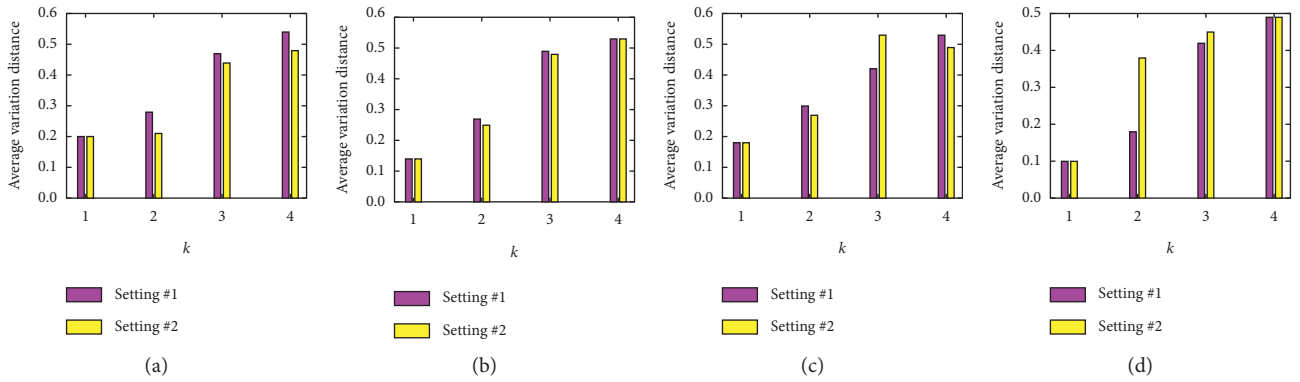


Figure 6: Accuracy vs. (k) ($\varepsilon_{\text{average}} = 2$). (a) Adult. (b) Abalone. (c) Bank marketing. (d) Car evaluation.

his data randomly and then perturbed the data with a total privacy budget $\varepsilon_{\text{total}} = 3 \times \varepsilon_{\text{average}} = 3 \times 2 = 6$. Then, the joint distribution of $k = 1, 2, 3, 4, 5$ attribute(s) is estimated.

Figure 7 compares the average computation time of two-dimensional data joint distribution estimation on the four real datasets with different average privacy budgets $\varepsilon_{\text{average}}$ and $k = 2$. As we can see, the computation takes only a few seconds. Moreover, when $\varepsilon_{\text{average}}$ is growing, the computation time increasing slowly even is unchanged. This is because the joint distribution is estimated by using LASSO regression, whose time complexity is mainly subject to the total number of records.

Table 2: Accuracy vs. k ($\varepsilon_{\text{average}} = 2$).

Datasets (settings)	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
Adult (S#1)	0.29	0.28	0.47	0.54	0.54
Adult (S#2)	0.29	0.27	0.43	0.48	0.51
Abalone (S#1)	0.36	0.36	0.42	0.53	0.53
Abalone (S#2)	0.36	0.36	0.39	0.45	0.49
Bank marketing (S#1)	0.36	0.36	0.42	0.48	0.53
Bank marketing (S#2)	0.36	0.36	0.41	0.43	0.45
Car evaluation (S#1)	0.27	0.34	0.40	0.49	0.52
Car evaluation (S#2)	0.27	0.34	0.37	0.41	0.47

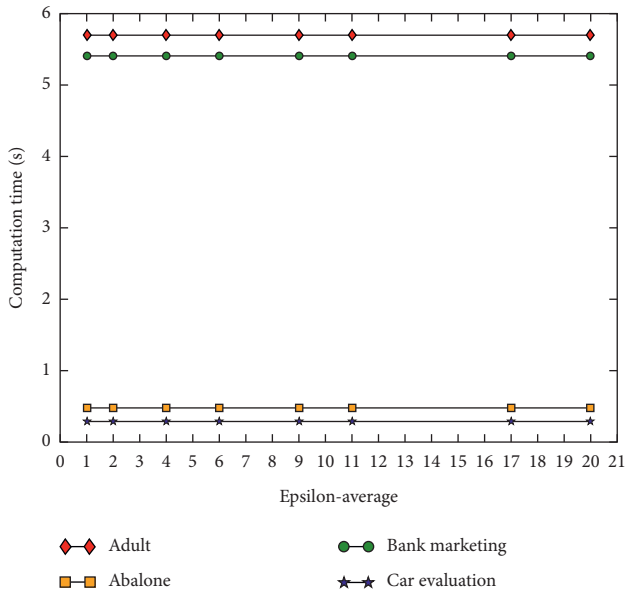


Figure 7: Computation time vs. $\varepsilon_{\text{average}}$ ($k = 2$).

6. Conclusion

In this paper, a new privacy notion PLDP is proposed to provide personalized privacy allocation under LDP. PLDP can be regarded as a generalized version of LDP. It allows users only to report their partial data and perturb them with distinct privacy budgets. Then, in order to estimate the joint distribution of multidimensional data, we develop an aggregation algorithm under PLDP, which is based on LASSO regression and information entropy. Finally, experiments on four real datasets validate the superiority and efficiency of PLDP, compared with the traditional LDP.

Data Availability

No data were used to support the findings of this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Jiangsu Basic Research Programs-Natural Science Foundation under grant no. BK20181407, in part by the National Natural Science Foundation of China under grant nos. U1936118 and 61672294, in part by Six Peak Talent Project of Jiangsu Province (R2016L13), Qinglan Project of Jiangsu Province, and "333" Project of Jiangsu Province, in part by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund, in part by the Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET) fund, China. Zhihua Xia was supported by BK21+ program from the Ministry of Education of Korea.

References

[1] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

[2] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proceedings of the 2013 IEEE 54th annual Symposium on Foundations of computer science*, pp. 429–438, IEEE, Berkeley, CA, USA, October 2013.

[3] A. G. Thakurta, A. H. Vyrros, U. S. Vaishampayan et al., "Emoji frequency detection and deep link frequency," U.S. Patent-9-705908, 2017.

[4] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3571–3580, Long Beach, CA, USA, December 2017.

[5] Q. Ye, H. Hu, X. Meng et al., "PrivKV: key-value data collection with local differential privacy," in *Proceedings of the2019 IEEE Symposium on Security and Privacy (SP)*, pp. 317–331, San Francisco, CA,USA, May 2019.

[6] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP)*, pp. 127–143, IEEE, San Francisco, CA,USA, May 2018.

[7] C. Dwork, "Differential Privacy," in *Proceedings of the33rd International Colloquium on Automata, Languages and Programming (ICALP)*, Venice, Italy, July 2006.

[8] Ú Erlingsson, V. Pihur, and A. Korolova, "Rappor: randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1054–1067, Scottsdale, AZ, USA, November 2014.

[9] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[10] R. Tibshirani, "Regression shrinkage and selection via the lasso: a retrospective," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 3, pp. 273–282, 2011.

[11] J. W. Kim, D.-H. Kim, and B. Jang, "Application of local differential privacy to collection of indoor positioning data," *IEEE Access*, vol. 6, pp. 4276–4286, 2018.

[12] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, pp. 127–135, Portland, OR, USA, June 2015.

[13] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete distribution estimation under local privacy," 2016, http://arxiv.org/abs/1602.07387.

[14] T. K. Ren, C.-M. Yu, W. Yu et al., "The expectation-maximization algorithm," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2151–2166, 2018.

[15] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[16] Z. Zhang, T. Wang, N. Li et al., "Calm:onsistent adaptive local marginal for marginal release under local differential privacy," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 212–229, Toronto, Canada, October 2018.

[17] S. Wang, L. Huang, M. Tian et al., "Personalized privacy-preserving data aggregation for histogram estimation," in *Proceedings of the2015 IEEE Global Communications*

Conference (GLOBECOM), pp. 1–6, Honolulu, HI, USA, November 2015.

[18] N. I. E. Yiwen, W. Yang, L. Huang et al., "A utility-optimized framework for personalized private histogram estimation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 655–669, 2018.

[19] R. Chen, H. Li, A. K. Qin et al., "Private spatial data aggregation in the local setting," in *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pp. 289–300, IEEE, Helsinki, Finland, May 2016.

[20] R. Sarathy and K. Muralidhar, "Evaluating Laplace noise addition to satisfy differential privacy for numeric data," *Transactions on Data Privacy*, vol. 4, no. 1, pp. 1–17, 2011.

[21] N. McSherry, D. J. au, and O. au, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis optimal differentially private mechanisms for randomised response," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, pp. 19–30, Providence, RI, USA, June 2009.

[22] N. Holohan, D. J. Leith, and O. Mason, "Optimal differentially private mechanisms for randomised response," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2726–2735, 2017.

[23] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2879–2887, Long Beach, CA, USA, January 2014.

[24] S. Wang, L. Huang, P. Wang et al., "Private weighted histogram aggregation in crowdsourcing," in *Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications*, pp. 250–261, Yellow Mountains, China, August 2016.

[25] E. Yilmaz, M. Al-Rubaie, and J. M. Chang, "Locally differentially private naive bayes classification," 2019, http://arxiv.org/abs/1905.01039.

[26] T. Wang, J. Blocki, N. Li et al., "Locally differentially private protocols for frequency estimation," in *Proceedings of the 26th USENIX Security Symposium (USENIX Security 17)*, pp. 729–745, Vancouver, BC, Canada, August 2017.

[27] P. J. Li, T. Wang, M. Lopuhaä-Zwakenberg et al., "Estimating numerical distributions under local differential privacy," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pp. 621–635, Portland, OR, USA, June 2020.

[28] P. J. Schweitzer, "Elementary symmetric polynomials," *Siam Review*, vol. 9, no. 3, pp. 590-591, 1967.

[29] D.-Y. Pan, Y. Fang, and E. au, "Maximum likelihood estimation growth," *Curve Models and Statistical Diagnostics*, pp. 77–158, Springer, Berlin, Germany, 2002.

[30] D. Y. Tsai, Y. Lee, and E. Matsuyama, "Information entropy measure for evaluation of image quality," *Journal of Digital Imaging*, vol. 21, no. 3, pp. 338–347, 2008.

[31] S. Dua, P. Graff, and P. au, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science., Irvine, CA, USA, 2019, http://archive.ics.uci.edu/ml.

[32] S. Moro, P. Cortez, and P. Rita, "A data-driven approach to predict the success of bank telemarketing," *Decision Support Systems*, vol. 62, pp. 22–31, 2014.

[33] R. Chen, Q. Xiao, Y. Zhang et al., "Differentially private high-dimensional data publication via sampling-based inference," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 129–138, Sydney, NSW, Australia, August 2015.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6662989, 7 pages
https://doi.org/10.1155/2021/6662989

Research Article

Weighted Nuclear Norm Minimization on Multimodality Clustering

Lei Du, Songsong Dai, Haifeng Song, Yuelong Chuang, and Yingying Xu

School of Electronics and Information Engineering, Taizhou University, Taizhou, China

Correspondence should be addressed to Haifeng Song; isshf@126.com

Received 7 December 2020; Revised 24 December 2020; Accepted 6 January 2021; Published 28 January 2021

Academic Editor: Liguo Zhang

Generally, multimodality data contain different potential information available and are capable of providing an enhanced analytical result compared to monosource data. The way to combine the data plays a crucial role in multimodality data analysis which is worth investigating. Multimodality clustering, which seeks a partition of the data in multiple views, has attracted considerable attention, for example, robust multiview spectral clustering (RMSC) explicitly handles the possible noise in the transition probability matrices associated with different views. Spectral clustering algorithm embeds the input data into a low-dimensional representation by dividing the clustering problem into k subproblems, and the corresponding eigenvalue reflects the loss of each subproblem. So, the eigenvalues of the Laplacian matrix should be treated differently, while RMSC regularizes each singular value equally when recovering the low-rank matrix. In this paper, we propose a multimodality clustering algorithm which recovers the low-rank matrix by weighted nuclear norm minimization. We also propose a method to evaluate the weight vector by learning a shared low-rank matrix. In our experiments, we use several real-world datasets to test our method, and experimental results show that the proposed method has a better performance than other baselines.

1. Introduction

Clustering, a task of partitioning data points into multiple clusters, is a fundamental research problem in data mining and machine intelligence. A series of algorithms have been proposed over the past decades [1–7]. One of the representative methods is spectral clustering, which has a lot of applications [8–11]. With the development of information and communication technologies, which led to data production in most areas, it is relatively easy to capture features from a given subject. So, it is necessary to design new pattern recognition methods to deal with views of the same subjects. For example, in multilingual information retrieval, the same document can be represented by different languages, and each language can be regarded as a view. These individual views can provide complementary information to each other which can lead to improved performance on the learning task. In the context of multimodality clustering, it seeks to get a better clustering performance by leveraging the information from multiple views.

Many multimodality clustering methods have been proposed in recent years. In general, there are three steps when clustering multiple data $X = [X^{(1)}, X^{(2)}, \ldots, X^{(m)}]$ [12]:

(1) Obtain a similarity matrix S^i from each view X^i, $(i = 1, 2, \ldots, m)$

(2) Compute a projection of each similarity matrix into a space suitable for clustering

(3) Produce a clustering assignment (i.e., K-means)

The main difference between the multimodality clustering methods lies in the step where the information is collapsed to produce a single new representation. The first category (information merges in Step 1) merges $S = [S^{(1)}, S^{(2)}, \ldots, S^{(m)}]$ to get a new similarity matrix. The method presented in [13] is a Markov chain method for the generalized normalized cut on multimodality data; the method described in [14] uses the philosophy of co-regularization to make the clustering in different views agree with

each other, as described in [15]; RMSC is a Markov-chain-based multimodality spectral clustering method via low-rank and sparse decomposition. The second category (information merges in Step 2) of methods merges the information to generate a compatible projection for all views. In [16], the authors used canonical correlation analysis to maximize the correlation of subjects across the projected views. In the third step, spectral clustering produces the assignment by K-means. The assignment is not stable for the randomness of K-means, so the third category learns a stable assignment. For example, ensemble clustering [17, 18] methods are designed to find a stable assignment.

The standard nuclear norm minimization regularizes each singular value equally to pursue the convexity of the loss function, while the singular values have different meanings and should be treated differently. Gu [19] proposed a weighted nuclear norm method and applied it to image denoising. The weight vector is evaluated by the singular values of image patches and the noise variance, but it is not useful for multimodality clustering.

In [15], we presented a Markov-chain-based multimodality spectral clustering method via low-rank and sparse decomposition. In this paper, as shown in Figure 1, we extend our previous study by applying the weighted nuclear norm to multimodality clustering and propose a method to evaluate the weight vector. The difference between them is that RMSC recovers the low-rank matrix $\widehat{P}$ by solving a nuclear norm minimization (NNM) problem, while the proposed method recovers that by solving a weighted nuclear norm minimization (WNNM) problem. For the experiments, we use several real-world datasets to test our method. Experimental results show that the proposed method has a better performance than other baselines.

This paper is organized as follows. Section 2 briefly describes the related work from which our method is based on. Section 3 describes the reason for using WNNM on multimodality clustering, defines our algorithm, and presents the optimization procedure. Section 4 presents the results of our method and other multimodality clustering methods. Section 5 outlines the main contributions of the work presented in this paper.

2. Related Work

To make this paper clear, Table 1 summarizes the symbols used in this paper.

2.1. Spectral Clustering. Finding good clusters has been a focus of considerable research in pattern recognition. Spectral clustering applies the spectral graph theory [20] which gives the conditions where a graph can be divided into several nonconnected subgraphs. The method embeds the input data into a low-dimensional representation and then applies K-means.

Here we give the framework of the spectral clustering algorithm [8, 21, 22] (Algorithm 1).

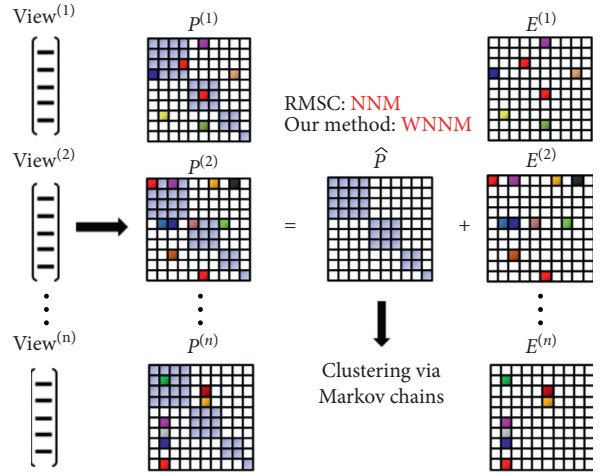


Figure 1: Framework of the robust multimodality on spectral clustering via low-rank and sparse decomposition; the proposed method is similar to RMSC [15]; the difference between them is that RMSC recovers the low-rank matrix $\widehat{P}$ by solving a nuclear norm minimization (NNM) problem, while the proposed method recovers that by solving a weighted nuclear norm minimization (WNNM) problem.

Table 1: Symbols used in this paper.

Symbol	Meaning
X	The whole dataset
$X^{(v)}$	The v-th view dataset
$S^{(v)}$	The v-th similarity matrix
$\widehat{P}$	Low-rank matrix
$P^{(i)}$	The v-th transition matrix
$E^{(i)}$	The v-th noise matrix
m	Number of views
n	Number of instances
λ_i	The i-th eigenvalue
σ_i	The i-th singular value
w_i	The i-th weight of σ_i
λ	Trade-off parameter

2.2. Robust Multimodality Spectral Clustering via Low-Rank and Sparse Decomposition (RMSC). Consider a set of multimodality data $X = [X^{(1)}, X^{(2)}, \ldots, X^{(m)}]$ with $X^{(i)} \in \mathbb{R}^{d^{(i)} \times n}$, where m is the number of views, n is the number of data points, $d^{(i)}$ represents the feature dimension of the i-th view, and the j-th column $X^{(i)}_{\cdot j}$ in $X^{(i)}$ represents the features of the j-th data point in the i-th view ($j = 1, 2, \ldots, n$; $i = 1, 2, \ldots, m$). The first step of RMSC is using Gaussian kernels to define the similarity matrix, i.e.,$S_{ij} = \exp(-\|x_i - x_j\|_2^2/\sigma^2)$ where $\|.\|_2$ denotes the ℓ_2 norm and σ^2 denotes the standard deviation (e.g., one can set σ^2 to be the average Euclidean distance over all pairs of data points). The second step is to construct the transition matrix P by $P = D^{-1}S$ where D is a diagonal matrix with $D_{ii} = d_i = \sum_{j=1}^{n} S_{ij}$. Under the low-rank and sparse assumptions, they formulate the transition matrix construction problem as

Input: $X \in \mathbb{R}^{n\times d}$
(1) Construct the similarity matrix S by Gaussian kernel, where S_{ij} represents the similarity of the i−th sample and the j−th sample.
(2) Compute the normalized symmetrical Laplacian $L = I - D^{-1/2}SD^{-1/2}$, where D is a diagonal matrix with $D_{ii} = \sum_j S_{ij}$.
(3) Let U be a matrix with columns representing the top k eigenvectors of L.
(4) Normalize each row of U.
(5) Run the k-means algorithm on U.
Output: the result of k-means.

Algorithm 1: Spectral clustering.

$$\begin{aligned} &\min_{\widehat{P},E^{(i)}} \operatorname{rank}(\widehat{P}) + \lambda \sum_{i=1}^{m} \left\| E^{(i)} \right\|_0, \\ &\text{s.t. } i = 1, 2, \ldots, m, \quad P^{(i)} = \widehat{P} + E^{(i)}, \widehat{P} \geq 0, \widehat{P}1 = 1, \end{aligned} \tag{1}$$

where the ℓ_0 norm $\|E^{(i)}\|_0$ is the number of nonzero elements in $E^{(i)}$, rank $(\widehat{P})$ represents the rank of $\widehat{P}$, 1 is a vector with all ones, and λ is a trade-off parameter. Note that the constraints $\widehat{P} \geq 0, \widehat{P}1 = 1$ enforce $\widehat{P}$ to be a transition probability matrix, i.e., each of its rows is a probability distribution.

As the problem is nonconvex, they replace rank $(\widehat{P})$ with the trace norm $\|\widehat{P}\|_*$, and $\|E^{(i)}\|_0$ with the ℓ_1 norm $\|E^{(i)}\|_1$, resulting in the following convex optimization problem:

$$\begin{aligned} &\min_{\widehat{P},E^{(i)}} \|\widehat{P}\|_* + \lambda \sum_{i=1}^{m} \left\| E^{(i)} \right\|_1 \\ &\text{s.t. } i = 1, 2, \ldots, m, \quad P^{(i)} = \widehat{P} + E^{(i)}, \widehat{P} \geq 0, \widehat{P}1 = 1. \end{aligned} \tag{2}$$

The ℓ_1 norm $\|E^{(i)}\|_1 = \sum_{(i,j)} |E_{ij}|$ is well known to be a convex surrogate of $\|E\|_0$. Then, they propose an optimization procedure to solve this problem via the augmented Lagrangian multiplier (ALM) scheme, which has shown its good balance between efficiency and accuracy in many matrix learning problems.

Let σ_i (Σ are in a nonascending order) represent the i-th singular value ($\Sigma = (\sigma_1, \sigma_2, \ldots)$). When updating Q, the subproblem is

$$\min_Q \|Q\|_* + \frac{\mu}{2} \left\| \widehat{P} - Q + \frac{Z}{\mu} \right\|_F^2. \tag{3}$$

Let $U\Sigma V^T$ be the SVD form of $(\widehat{P} + t(Z/\mu))$, and the solution is as follows:

$$Q = U \mathcal{S}_{1/\mu}(\Sigma) V^T, \tag{4}$$

where $\mathcal{S}_\delta(\sigma_i) = \max(\sigma_i - \delta, 0) + \min(\sigma_i + \delta, 0)$ is the shrinkage operator.

In the optimization procedure, each single value adds or subtracts the same value. So, RMSC treats each singular value equally, which may degrade the performance of the result of clustering.

2.3. Weighted Nuclear Norm. Gu et al. [19] studied the weighted nuclear norm minimization (WNNM) problem, where the singular values are assigned different weights. The definition of weighted nuclear norm of a matrix X is as follows:

$$\|X\|_{w,*} = \Sigma_i \left| w_i \sigma_i(X) \right|_1. \tag{5}$$

They analyzed the solutions of the WNNM problem under different weight conditions and proposed a method to evaluate the weight vector according to many image patches when applied the WNNM algorithm to image denoising. The difference between WNNM and our method is as follows: (1) we extend the weighted nuclear norm to multimodality clustering; (2) the methods which evaluated the weight vector were different; the former evaluates the weight vector according to image patches, while our method evaluates that by matrix decomposition.

3. The Proposed Method: Weighted RMSC

In this section, we present how to apply the weighted nuclear norm to multimodality clustering.

As described in Section 2.2, RMSC treats each singular value when updating $Q(\widehat{P} = tQ)$, while for spectral clustering, different eigenvalues of L have different meaning.

According to [23], the RatioCut object function is defined as

$$\begin{aligned} \operatorname{RatioCut}(A_1, A_2, \ldots, A_k) &= \sum_{i=1}^{k} h_i^T L h_i \\ &= \sum_{i=1}^{k} \left(H^T L H \right)_{ii} = \operatorname{Tr}\left(H^T L H \right) \\ \text{s.t. } H^T H &= I, \end{aligned} \tag{6}$$

where Tr (.) denotes the trace of a matrix and h_i represents the i-th column of H.

So, the RatioCut object function is defined as

$$\begin{aligned} &\min_H \operatorname{Tr}\left(H^T L H \right) \\ &\text{s.t. } H^T H = I. \end{aligned} \tag{7}$$

According to Rayleigh–Ritz theorem [24] the problem has a fixed solution and H is constructed by the top k eigenvectors of L. From equation (6), we can find that the normalized spectral clustering divides the problem into k subproblems. Each subproblem partitions the points into 2 clusters, and h_i ($i = 2, \ldots, k$) is the solution of the subproblem (h_1 is an all-one vector, which represents it dividing all the points into the same cluster; this partition is useless).

As $H^T H = I$, the RatioCut object function can be rewritten as

$$\begin{aligned} &\text{RatioCut}\left(A_1, A_2, \ldots, A_k\right) = \sum_{i=1}^{k} h_i^T L h_i = \sum_{i=1}^{k} \lambda_i \\ &\text{s.t. } H^T H = I. \end{aligned} \tag{8}$$

So, we can find that the loss of each subproblem has a relation with the corresponding eigenvalue; the small eigenvalue reflects the little loss of the subproblem, so the smaller the eigenvalue λ is, the larger the weight of corresponding eigenvector should be assigned. It is also known that

$$L = I - \widehat{P}. \tag{9}$$

So, the larger eigenvalues (or singular values, $\sigma_i = \lambda_i$) of $\widehat{P}$ are more important than the smaller ones when updating $\widehat{P}$ in RMSC; the larger the eigenvalues, the less they should be shrunk. Therefore, the weight assigned to σ_i should be inversely proportional to σ_i. We let

$$w_i = \frac{c}{\left(\sigma_i + \epsilon\right)}, \tag{10}$$

where $c > 0$ is a constant; $\epsilon = 10^{-16}$ is to avoid dividing by zero.

For multimodality clustering, we can construct m Laplacian matrices, leading to m groups of σ_i, which are not equal. So evaluating accurate σ_i is a challenging procedure, leading to the difficulty to make sure the weight of singular value. As we know, the output of RMSC is a shared low-rank matrix, and all the views share the same singular values. So, one way to evaluate the singular values is making use of other multimodality clustering algorithms, such as RMSC, to get the shared Laplacian matrix and then evaluating the final Σ by the singular values of the shared Laplacian matrix.

Following RMSC, under the low-rank and sparse assumptions, we formulate the transition matrix construction problem as follows:

$$\begin{aligned} &\min_{\widehat{P}, E^{(i)}} \|\widehat{P}\|_{w,*} + \lambda \sum_{i=1}^{m} \left\|E^{(i)}\right\|_1 \\ &\text{s.t. } i = 1, 2, \ldots, m, \quad P^{(i)} = \widehat{P} + E^{(i)}, \widehat{P} \geq 0, \widehat{P}1 = 1. \end{aligned} \tag{11}$$

The optimization problem (11) is still challenging because the matrix $\widehat{P}$ has two constraints. We introduce an auxiliary variable Q to solve this problem. The optimization problem (11) becomes as follows:

$$\begin{aligned} &\min_{\widehat{P}, Q, E^{(i)}} \|Q\|_{w,*} + \lambda \sum_{i=1}^{m} \left\|E^{(i)}\right\|_1 \\ &\text{s.t. } i = 1, 2, \ldots, m, \quad P^{(i)} = \widehat{P} + E^{(i)}, \\ &\qquad \widehat{P} \geq 0, \quad \widehat{P}1 = 1, \widehat{P} = Q. \end{aligned} \tag{12}$$

The corresponding augmented Lagrange function of (12) is

$$\begin{aligned} \mathscr{L}\left(\widehat{P}, tQn, qE^{(i)}\right) &= \|Q\|_{w,*} + \lambda \sum_{i=1}^{m} \left\|E^{(i)}\right\|_1 \\ &+ \sum_{i=1}^{m} \langle Y^{(i)}, \widehat{P} + E^{(i)} - P^{(i)} \rangle \\ &+ \frac{\mu}{2} \sum_{i=1}^{m} \left\|\widehat{P} + E^{(i)} - P^{(i)}\right\|_F^2 \\ &+ \langle Z, \widehat{P} - Q \rangle + \frac{\mu}{2} \|\widehat{P} - Q\|_F^2 \quad \text{s.t. } \widehat{P} \geq 0, \widehat{P}1 = 1, \end{aligned} \tag{13}$$

where Z, $Y^{(i)}$ represent the Lagrange multipliers, $\langle \cdot, \cdot \rangle$ denotes the inner product of matrices (i.e., for two matrices A and B, $\langle A, B \rangle = A^T B$), and $\mu > 0$ is an adaptive penalty parameter.

The sketch of the proposed algorithm is shown in Algorithm 2. Next we will present the update rules for each of $\widehat{P}$, Q, and $E^{(i)}$.

When other variables are fixed, the subproblem with respect to Q is

$$\min_Q \|Q\|_{w,*} + \frac{\mu}{2} \left\|\widehat{P} - Q + \frac{Z}{\mu}\right\|_F^2. \tag{14}$$

More specifically, let $U \Sigma V^T$ be the SVD form of $(\widehat{P} + t\,(Z/\mu))$. We use RMSC to evaluate the final Σ and use it to evaluate W via Equation (10). According to [19], the solution to (14) is as follows:

$$Q = U \mathcal{S}_W (\Sigma) V^T. \tag{15}$$

The subproblem with respect to $E^{(i)}$ $(i = 1, 2, \ldots, m)$ can be simplified as

$$\min_{E^{(i)}} \lambda \left\|E^{(i)}\right\|_1 + \frac{\mu}{2} \left\|E^{(i)} - \left(P^{(i)} - \widehat{P} - \frac{Y^{(i)}}{\mu}\right)\right\|_F^2, \tag{16}$$

which has a closed form solution $E^{(i)} = \mathcal{S}_{\lambda/\mu}(P^{(i)} - \widehat{P} - (Y^{(i)}/\mu))$.

With other variables being fixed, we update $\widehat{P}$ by solving

$$\begin{aligned} \widehat{P} = \text{argmin}_{\widehat{P}} &\frac{\mu}{2} \sum_{i=1}^{m} \left\|\widehat{P} + E^{(i)} - P^{(i)} + \frac{Y^{(i)}}{\mu}\right\|_F^2 \\ &+ \frac{\mu}{2} \left\|\widehat{P} - Q + \frac{Z}{\mu}\right\|_F^2 \quad \text{s.t. } \widehat{P} \geq 0, \widehat{P}1 = 1. \end{aligned} \tag{17}$$

The solution is given by RMSC, which can be decomposed into n independent subproblems. Each subproblem is a proximal operator problem with probabilistic simplex constraint, which can be efficiently solved by the projection algorithm.

4. Experimental Setup

The proposed method was tested on several real-world datasets; the details are shown in Table 2.

In all the experiments, we use six metrics to measure the clustering performances: F-score, precision,

Input: $\lambda, P^{(i)} \in \mathbb{R}^{n\times n}$ $(i = 1, 2, \ldots, m)$.
Initialize: $\widehat{P} = 0, Q = 0, Z = 0, Y^{(i)} = 0, E^{(i)} = 0, \mu = 10^{-6}, \rho = 1.9$, $\max_\mu = 10^{10}, \epsilon = 10^{-8}$.
Evaluate W by running RMSC.
Repeat
(1) Let $C \longleftarrow (1/(m+1))(Q - (Z/\mu) + \sum_{i=1}^{m}(P^{(i)} - E^{(i)} - (Y^{(i)}/\mu)))$.
(2) **For** $j = 1, 2, \ldots, n$
  Update $\widehat{P}_j$
(3) **For** $i = 1, 2, \ldots, m$
  Update $E^{(i)}$ via equation (16).
(4) Update Q via equation (15).
(5) Set $Z \longleftarrow Z + \mu(\widehat{P} - tQ)$.
(6) **For** $i = 1, 2, \ldots, m$
  Set $Y^{(i)} \longleftarrow Y^{(i)} + \mu(\widehat{P} + E^{(i)} - P^{(i)})$.
(7) Set $\mu \longleftarrow \min(\rho\mu, \max_\mu)$.
Until $\min(\|\widehat{P} + E^{(i)} - P^{(i)}\|_\infty, \|\widehat{P} - Q\|_\infty) \le \epsilon$.
Output: $\widehat{P}$, $E^{(i)}$ $(i = 1, 2, \ldots, m)$.

Algorithm 2: Weighted nuclear norm on robust multimodality clustering.

Table 2: Statistics of the real-world datasets.

Dataset	Views	Instances	Clusters
BBCSports	2	544	5
WebKb	2	1051	2
Reuters	5	600	6
UCI	3	2000	10

recall, normalized mutual information (NMI) [25], entropy, and adjusted rand index (Adj-RI) [26]. Note that higher values indicate better performance except for entropy.

When evaluating the weight vector, there is a constant parameter c. We set $c = 0.00001$ in all the experiments. Similarity matrices are constructed by Gaussian kernels. σ^2 is set to the median of the Euclidean distance between every pair of data points for all of the datasets except BBCSports ($\sigma^2 = 100$). λ is set to be 0.005.

5. Experimental Results

We chose the following six multimodality clustering algorithms as baselines:

(1) Single view: performing spectral clustering on a single view.
(2) Feature concatenation: concatenating all the features of each view and then performing spectral clustering on the new representation.
(3) Kernel addition: constructing the similarity matrices from each view and then averaging all the matrices to obtain a new similarity matrix.
(4) Mixture of Markov chains (MMC): a mixture of Markov chains defined on each view [13].
(5) Co-regularized spectral clustering (Co-Reg): making use of the philosophy of co-regularization to make the clustering in different views agree with each other [14].
(6) Robust multiview spectral clustering via low-rank and sparse decomposition (RMSC): a Markov-chain-based multimodality spectral clustering method via low-rank and sparse decomposition [15].

Following the settings in [14], we use the Gaussian kernel to construct similarity matrix for each view if needed in all algorithms.

Table 3 shows the results of the proposed method and the baselines on BBCSports. As can be seen, the proposed method shows superior performance gains over the baselines with respect to all the six metrics. Here are some statistics: the results of our method indicate a relative increase of 4.82%, 2.88%, 2.10%, and 6.58% with respect to F-score, precision, NMI, and Adj-RI, respectively, compared to the corresponding second best baseline.

Table 4 shows the results of the proposed method and the baselines on UCI. As can be seen, the proposed method shows superior performance gains over the baselines with respect to all the six metrics. Here are some statistics: the results of our method indicate a relative increase of 1.35%, 1.62%, 1.32%, and 1.647.59% with respect to F-score, precision, NMI, and Adj-RI, respectively, compared to the corresponding second best baseline.

Table 5 shows the results of the proposed method and the baselines on WebKb. As can be seen, the proposed method shows superior performance gains over the baselines with respect to most of the six metrics. Here are some statistics: the results of our method indicate a relative increase of 0.32%, 2.46%, and 0.82% with respect to F-score, NMI, and Adj-RI, respectively, compared to the corresponding second best baseline. Although the precision value of the proposed method is lower than that of kernel addition, the difference is small.

Table 6 shows the results of the proposed method and the baselines on Reuters. As can be seen, the proposed method shows superior performance gains over the baselines with respect to all the six metrics. Here are some statistics: the results of our method indicate a relative increase of 1.89%, 3.56%, 5.85%, and 4.74% with respect to

Table 3: Comparison results on BBCSports.

Method	*F*-score	Precision	Recall	NMI	Entropy	Adj-RI
SC view 1	0.767 (0.003)	0.785 (0.011)	0.750 (0.017)	0.717 (0.004)	0.609 (0.021)	0.696 (0.002)
SC view 2	0.410 (0.008)	0.319 (0.028)	0.589 (0.066)	0.218 (0.020)	1.772 (0.060)	0.146 (0.038)
Feature concat	0.669 (0.019)	0.654 (0.017)	0.688 (0.049)	0.618 (0.023)	0.853 (0.039)	0.563 (0.020)
Kernel addition	0.672 (0.021)	0.657 (0.022)	0.691 (0.049)	0.621 (0.029)	0.847 (0.063)	0.566 (0.024)
MMC	0.788 (0.000)	0.825 (0.000)	0.755 (0.000)	0.728 (0.000)	0.571 (0.000)	0.726 (0.000)
Co-Reg	0.766 (0.763)	0.788 (0.780)	0.745 (0.747)	0.718 (0.712)	0.603 (0.621)	0.695 (0.691)
RMSC	0.851 (0.056)	0.868 (0.027)	0.836 (0.081)	0.810 (0.026)	0.408 (0.034)	0.806 (0.070)
Ours	**0.892 (0.003)**	**0.893 (0.005)**	**0.891 (0.002)**	**0.827 (0.004)**	**0.380 (0.011)**	**0.859 (0.004)**

Table 4: Comparison results on UCI.

Method	*F*-score	Precision	Recall	NMI	Entropy	Adj-RI
SC view 1	0.574 (0.029)	0.563 (0.028)	0.585 (0.032)	0.632 (0.019)	1.231 (0.063)	0.526 (0.033)
SC view 2	0.583 (0.023)	0.574 (0.028)	0.591 (0.018)	0.643 (0.013)	1.191 (0.045)	0.536 (0.026)
SC view 3	0.377 (0.004)	0.364 (0.011)	0.392 (0.009)	0.483 (0.005)	1.732 (0.024)	0.306 (0.006)
Feature concat	0.448 (0.014)	0.433 (0.020)	0.466 (0.013)	0.552 (0.016)	1.506 (0.058)	0.385 (0.017)
Kernel addition	0.746 (0.024)	0.723 (0.040)	0.773 (0.015)	0.784 (0.014)	0.736 (0.055)	0.717 (0.027)
MMC	0.729 (0.052)	0.708 (0.066)	0.752 (0.040)	0.771 (0.032)	0.778 (0.116)	0.697 (0.059)
Co-Reg	0.594 (0.650)	0.585 (0.634)	0.603 (0.668)	0.644 (0.687)	1.189 (1.051)	0.548 (0.610)
RMSC	0.816 (0.061)	0.805 (0.073)	0.828 (0.049)	0.832 (0.038)	0.565 (0.136)	0.795 (0.069)
Ours	**0.827 (0.041)**	**0.818 (0.052)**	**0.836 (0.029)**	**0.843 (0.022)**	**0.528 (0.082)**	**0.808 (0.046)**

Table 5: Comparison results on WebKb.

Method	*F*-score	Precision	Recall	NMI	Entropy	Adj-RI
SC view 1	0.592 (0.000)	0.645 (0.000)	0.546 (0.000)	0.167 (0.000)	0.618 (0.000)	0.028 (0.000)
SC view 2	0.889 (0.000)	0.824 (0.000)	0.965 (0.000)	0.532 (0.000)	0.406 (0.000)	0.618 (0.000)
Feature concat	0.896 (0.000)	0.836 (0.000)	0.966 (0.000)	0.559 (0.000)	0.384 (0.000)	0.648 (0.000)
Kernel addition	0.947 (0.000)	**0.947 (0.000)**	0.947 (0.000)	0.718 (0.000)	0.214 (0.000)	0.845 (0.000)
MMC	0.372 (0.017)	0.338 (0.017)	0.416 (0.029)	0.314 (0.025)	1.801 (0.063)	0.232 (0.020)
Co-Reg	0.365 (0.365)	0.334 (0.332)	0.405 (0.406)	0.312 (0.312)	1.804 (1.803)	0.225 (0.223)
RMSC	0.951 (0.000)	0.946 (0.000)	0.957 (0.000)	0.734 (0.000)	0.206 (0.000)	0.856 (0.000)
Ours	**0.954 (0.000)**	0.940 (0.000)	**0.970 (0.000)**	**0.752 (0.000)**	**0.201 (0.000)**	**0.863 (0.000)**

Table 6: Comparison results on Reuters.

Method	*F*-score	Precision	Recall	NMI	Entropy	Adj-RI
SC view 1	0.368 (0.017)	0.337 (0.009)	0.409 (0.042)	0.318 (0.017)	1.792 (0.031)	0.229 (0.014)
SC view 2	0.361 (0.016)	0.315 (0.019)	0.426 (0.030)	0.315 (0.028)	1.811 (0.070)	0.211 (0.021)
SC view 3	0.341 (0.005)	0.306 (0.008)	0.385 (0.012)	0.279 (0.015)	1.888 (0.039)	0.192 (0.007)
SC view 4	0.345 (0.014)	0.307 (0.014)	0.394 (0.014)	0.271 (0.016)	1.910 (0.041)	0.196 (0.018)
SC view 5	0.346 (0.014)	0.308 (0.016)	0.397 (0.022)	0.269 (0.017)	1.916 (0.042)	0.197 (0.018)
Feature concat	0.367 (0.014)	0.327 (0.013)	**0.419 (0.028)**	0.316 (0.023)	1.799 (0.055)	0.222 (0.016)
Kernel addition	0.371 (0.019)	0.333 (0.017)	**0.419 (0.027)**	0.313 (0.028)	1.806 (0.070)	0.228 (0.023)
MMC	0.367 (0.016)	0.334 (0.018)	0.408 (0.021)	0.308 (0.019)	1.812 (0.048)	0.227 (0.020)
Co-Reg	0.367 (0.016)	0.334 (0.018)	0.408 (0.021)	0.308 (0.019)	1.812 (0.048)	0.227 (0.020)
RMSC	0.372 (0.019)	0.338 (0.017)	0.415 (0.022)	0.325 (0.021)	1.770 (0.053)	0.232 (0.023)
Ours	**0.379 (0.021)**	**0.350 (0.009)**	0.413 (0.011)	**0.344 (0.021)**	**1.719 (0.015)**	**0.243 (0.016)**

F-score, precision, NMI, and Adj-RI, respectively, compared to the corresponding second best baseline. Although the recall value of the proposed method is lower than that of kernel addition and feature concatenation, the difference is small.

6. Conclusion

With the development of information and communication technologies, it is necessary to design new pattern recognition methods to deal with views of the same subjects. It is a

challenge task to deal with multimodality problems. Inspired by the previous work, we proposed a method applying the weighted nuclear norm to RMSC and gave a method to evaluate the weight vector, which distinguishes different single values. To solve the optimization problem, we designed a procedure based on ALM. To evaluate the proposed method, we apply it to four real-world datasets. Experimental results show that the proposed method has a superior performance than other baselines. In the future, we will continue the studies in multimodality clustering, including evaluating the weight vector more accurately and clustering on the large-scale datasets.

Data Availability

The BBCSports dataset used to support the findings of this study has been deposited in the UCD repository (http://mlg.ucd.ie/datasets/bbc.html) and the other datasets have been deposited in the UCI repository (http://archive.ics.uci.edu/ml/index.php).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the Cultivating Science Foundation of Taizhou University (2019PY014), the Agricultural Science and Technology Project of Taizhou (20ny13), the Zhejiang Provincial Natural Science Foundation of China (LQ21F020001 and LQ21A010001), and Taizhou Science and Technology Project (1901gy20).

References

[1] X.-H. Kuang, L. Liu, Q. Liu, and X. Li, "A clustering approach based on convergence degree chain for wireless sensor networks," *Security and Communication Networks*, vol. 8, no. 10, pp. 1878–1889, 2015.

[2] M. A. Azad, R. Morla, J. Arshad, and K. Salah, "Clustering voip caller for spit identification," *Security and Communication Networks*, vol. 9, no. 18, pp. 4827–4838, 2016.

[3] "Clustering and splitting charging algorithms for large scaled wireless rechargeable sensor networks," *Journal of Systems and Software*, vol. 113, pp. 381–394, 2016.

[4] M.-C. Su and C.-H. Chou, "A modified version of the *k*-means algorithm with a distance based on cluster symmetry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 674–680, 2001.

[5] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, 2002.

[6] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, John Wiley & Sons, Hoboken, NJ, USA, 2009.

[7] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[8] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[9] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition CVPR 2009*, pp. 2790–2797, Miami, FL, USA, June 2009.

[10] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10*, pp. 663–670, Haifa, Israel, June 2010.

[11] J. Feng, Z. Lin, H. Xu, and S. Yan, "Robust subspace segmentation with block-diagonal prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3818–3825, Seattle, WA, USA, June 2014.

[12] S. Kanaan-Izquierdo, A. Ziyatdinov, and A. Perera-Lluna, "Multiview and multifeature spectral clustering using common eigenvectors," *Pattern Recognition Letters*, vol. 102, pp. 30–36, 2018.

[13] D. Zhou and C. J. Burges, "Spectral clustering and transductive learning with multiple views," in *Proceedings of the 24th International Conference on Machine Learning, ACM*, pp. 1159–1166, 2007.

[14] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1413–1421, Lake Tahoe, NV, USA, December 2011.

[15] R. Xia, Y. Pan, L. Du, and J. Yin, "Robust multi-view spectral clustering via low-rank and sparse decomposition," in *Proceedings of the AAAI*, pp. 2149–2155, New York, NY, USA, February 2014.

[16] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 129–136, ACM, Montreal, Quebec, Canada, June 2009.

[17] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 3, pp. 337–372, 2011.

[18] X. Cai, F. Nie, and H. Huang, "Multi-view k-means clustering on big data," in *Proceedings of the IJCAI*, pp. 2598–2604, New York, NY, USA, August 2013.

[19] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2862–2869, Seattle, WA, USA, June 2014.

[20] S. Butler and F. Chung, "Spectral graph theory," *Handbook of linear algebra*, p. 47, CRC Press, Boca Raton, FL, USA, 2006.

[21] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[22] A. Y. Ng, M. I. Jordan, Y. Weiss et al., "On spectral clustering: analysis and an algorithm," *NIPS*, vol. 14, pp. 849–856, 2001.

[23] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[24] H. Lutkepohl, "Handbook of matrices," *Computational Statistics and Data Analysis*, vol. 2, no. 25, p. 243, 1997.

[25] H.-C. Huang, Y.-Y. Chuang, and C.-S. Chen, "Multiple kernel fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 1, pp. 120–134, 2012.

[26] Y. Zhao and G. Karypis, "Empirical and theoretical comparisons of selected criterion functions for document clustering," *Machine Learning*, vol. 55, no. 3, pp. 311–331, 2004.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 6662166, 14 pages
https://doi.org/10.1155/2020/6662166

Research Article

Detecting Web Spam Based on Novel Features from Web Page Source Code

Jiayong Liu,[1] Yu Su,[1] Shun Lv,[2] and Cheng Huang[1]

[1]*College of Cybersecurity, Sichuan University, Chengdu, China*
[2]*College of Computer Science, Sichuan University, Chengdu, Sichuan, China*

Correspondence should be addressed to Cheng Huang; opcodesec@gmail.com

Received 1 November 2020; Revised 24 November 2020; Accepted 4 December 2020; Published 17 December 2020

Academic Editor: Liguo Zhang

Search engine is critical in people's daily life because it determines the information quality people obtain through searching. Fierce competition for the ranking in search engines is not conducive to both users and search engines. Existing research mainly studies the content and links of websites. However, none of these techniques focused on semantic analysis of link and anchor text for detection. In this paper, we propose a web spam detection method by extracting novel feature sets from the homepage source code and choosing the random forest (RF) as the classifier. The novel feature sets are extracted from the homepage's links, hypertext markup language (HTML) structure, and semantic similarity of content. We conduct experiments on the WEBSPAM-UK2007 and UK-2011 dataset using a five-fold cross-validation method. Besides, we design three sets of experiments to evaluate the performance of the proposed method. The proposed method with novel feature sets is compared with different indicators and has better performance than other methods with a precision of 0.929 and a recall of 0.930. Experiment results show that the proposed model could effectively detect web spam.

1. Introduction

With the rapid development of the network, web applications are becoming more and more popular in the recent years, among which search engines are one of the most common web tools for people to gain information every day [1]. As the most popular search engine worldwide, Google processes over 40,000 search queries every second on average, which translates to over 3.5 billion searches per day and 1.2 trillion searches per year worldwide in 2012 [2]. There are data [3, 4] indicating that 85% of Internet users find websites through search engines and 90% of Internet users do not go past the first three pages on search results. Spammers design pages delicately to improve rankings as most users only access the first page of search results. There has been a brief definition of web spamming in the literature [5]; shortly speaking, web spamming is a black-hat search engine optimization (SEO) that deceive search engines to increase the ranking of a page in search engine results. These web pages are called web spam. As evident, spammers try to deceive search engines and attract end users to click on web spam sites. They not only reduce the effectiveness and efficiency of search engine results since web spam pages take much time to process but may also be full of malicious content and links. Lina et al. [6] present threats and related attacks about web spam. Although search engine companies have utilized various methods to counter spam [7], it is still a challenge to prevent the increase of black-hat SEO technology and the growth of spam pages nowadays. Therefore, it is of great significance to detect web spam with efficiency and accuracy.

Many researchers and experts have conducted much research on spam in this field. Several researchers have relied on feature extraction from text and links on the web page [8, 9]. Some other researchers detect web spam by crawlers observing different versions of the web page returned to search engines and ordinary users [10, 11], as well as there are methods based on spam purposes and user access logs [12]. Detection methods have also evolved continuously from the original statistical characteristics to determine whether a web page is spam to automatic monitoring using machine learning and deep learning, and the efficiency and

accuracy of detection are also continuously improved. We are motivated by previous work in the field of web spam detection and cybersecurity, which has proved the viability of web spam detection using a combination of machine learning and effective features. We use similar insights to support the discovery of web spam based on novel features. Our method is different from that in previous studies in that we extract not only link-based statistical features but also semantic features based on text content analysis and structural features based on the structure of web pages from the source code. Additionally, in terms of real-world aerial applications, the proposed method could be deployed in the browser. For example, the judgment of each web page in the users' search results by the proposed method can provide constructive conclusions to users and browser manufacturers. As the proposed method has features based on semantics, it is helpful to detect spam in web pages where links to spam content are easily injected into.

In this paper, we proposed a method using machine learning algorithm RF that combines feature extraction and feature selection to classify whether a web page is spam or not. Note that, for a binary classification problem, the classifier aims to distinguish the web page as spam or nonspam. The main contributions of this paper are as follows:

(1) This paper considers some previously undescribed features for web spam detection. We extract three novel feature subsets by studying homepage's links, texts, and structure based on statistical and semantic similarity analysis. The experimental results prove that the importance of novel features ranks high and effective.

(2) This paper applies a feature selection method for precomputed features related to the homepage to reduce computational consumption and improve accuracy. We introduce random forest algorithm for building the web spam detection model. The method could automatically distinguish web spam and a normal page from the website homepage.

(3) We evaluate the proposed method with comprehensive evaluation metrics for binary classification problems. Our method achieves the F1 score of 92.9%, which is higher than that of the existing methods. The experimental results show that our method can effectively detect web spam.

The rest of this paper is organized as follows. Section 2 presents related work regarding web spam detection. Section 3 describes the proposed approach in detail, and Section 4 evaluates the proposed method and the results of our experiments. Finally, we discuss our conclusions and future work.

2. Related Work

Web spam is often categorized into four classes: content spam, link spam, cloaking, and redirection. Several researchers and experts present kinds of methods to combat web spam correspondingly.

There were many research methods from different perspectives in the early stages. For example, Jakub Piskorski et al. [13] explored linguistic features focused on the utility of content-based linguistic features with computing 208 linguistic attributes, Benczúr et al. [14] conducted commercial intent analysis because other than the ordinary methods depending on the website itself; the authors thought much web spam was for commercial purpose, Bíró et al. [15] applied an extension of latent Dirichlet allocation (LDA) which is a linked LDA technique for web spam classification since topics are propagated along with links in such a way that the linked document directly influences the words in the linking document, and Liu et al. [16] analyzed web spam with user behavior where user visiting patterns of spam pages and three user behavior features are proposed to separate web spam from ordinary ones. Luca et al. [17] studied the spectrum of black-hat cloaking techniques that target browser, network, or contextual cues to detect organic visitors. Their anticloaking system can detect whether a web page would split view content returned to two or more distinct browsing profiles.

With machine learning developing by, a plurality of popular machine learning algorithms combined with sorts of feature engineering methods are applied to detect web spam. Machine learning techniques are more flexible than other methods; some difficult problems can be solved and more accuracy becomes a reality. Liu et al. [18] used a sentiment analysis model based on topic enhanced word embedding to obtain more complete text context information. The document topic distribution matrix is used to extract the document features. Reza Mohammadi et al. [19] proposed a method to improve support vector machine (SVM) algorithm by using two nonlinear kernels in twin support vector machine (MKTWSVM), which was experimented on both UK-2006 and UK-2007 datasets. The authors used a language-model approach and qualified-link analysis on detection. Fdez-Glez et al. [20] proposed a new framework according to combine different techniques, particularly suitable for filtering spam content on web pages. Mei et al. [21] proposed an improved PageRank algorithm based on web page differentiation (DPR), which evaluates pages authority according to its links' numbers and assigns corresponding weights according to its authoritativeness when assigning PageRank values. They combined DPR with K-means, designed a differentiation page-based K-means algorithm. Jelodar et al. [22] presented a systematic framework based on the chi-squared automatic interaction detector algorithm and a modified string matching algorithm. The author used the modified knuth–morris–pratt algorithm to extract features from Alexa Top 500 Global Sites and Bing search engine results in 500 queries; then, they generated a tree model with useful attributes that can detect web spam. Asdaghi and Soleimani [23] proposed a new backward elimination feature selection approach with the Naive Bayes (NB) classifier.

Many experts also used different neural networks and deep learning algorithms to detect web spam. Renato Moraes et al. [24] presented a performance evaluation of different models of artificial neural networks used to automatically

classify and filter real samples of web spam based on their contents. Li et al. [25] introduced the deep belief networks and combined with the synthetic minority oversampling technique (SMOTE) and denoising autoencoder (DAE) algorithm to improve the classification performance of web spam. In [26], the authors presented a framework called FS2RNN, a feature selection scheme using recurrent neural networks (RNNs), for the classification of spam nodes. In this framework, the dataset is preprocessed before applying RNNs in which principal component analysis (PCA) is used for dimension reduction on the dataset and recursive feature elimination (RFE) is used for feature selection. Belahcen et al. [27] addressed the web spam detection problem by using the graph neural network (GNN) architecture, which can act as a mixed transductive-inductive model that is able to classify pages by using both the explicit memory of the classes assigned to the training examples and the information stored in the network parameters.

In addition to detecting traditional e-mail spam and web spam, there are many scholars studying spam in social media called social spam, such as spam based on blogs, tweets, and YouTube videos. Fu et al. [28] presented a framework detecting spammers by measuring how careful a user is when she is about to follow a potential spammer. Samsudin et al. [29] proposed a framework that extracted features by using data collected from the YouTube spam dataset to detect YouTube comments spam. To deal with users who are affected by social spam, Ezpeleta et al. [30] focused on mood analysis and all content-based analysis techniques. Based on these heuristic research studies, we can apply to the problem that needs to be solved in this paper.

3. Proposed Method

In this section, we discuss the proposed method framework, give a comprehensive process of mining novel features, and determine classification algorithm training for the detection of the web spam model presented in this paper. The framework of the proposed method is depicted in Figure 1. The input is the web pages of sorts of websites. The output is a list of web pages with predicted classification scores, where a higher score indicates that the web page is more likely to be web spam. The proposed method is composed of 3 components: the prepossessing, features, and detection model. The number in brackets is the number of features. Next, we will describe the functionality and specific implementation methods of each component in detail.

3.1. Data Augmentation. We design our method based on the WEBSPAM-UK2007 dataset [31]. In the labeled samples given by the original dataset, the proportion of spam is only 6%. One of the main challenges we face is that the data are very imbalanced. There is no doubt that machine learning algorithms are data-driven approaches. It means that the performance of the model is highly related to data. Therefore, to augment the data, we extract more original data from the results of previous studies on this dataset. The summary of the augmented data method is to select the labeled data from the labeling results with high accuracy of the previous studies on the dataset, which are used as the labels of our data. Detailed information about data augmentation is given in Section 4.1.

3.2. WARC Parser. First of all, the dataset is structured, but the complex structured data cannot be directly applied because it contains unnecessary information. We need to process raw data. The original web pages' HTML documents in each host are arranged in sequence and stored in separate Web Archive (WARC) format proposed by the Internet Archive. The WARC format is an extension of the ARC File Format that has traditionally been used to store "web crawls" as sequences of content blocks harvested from the World Wide Web. Figure 2 shows a code snippet of a WARC file. We can see that each capture in a WARC file is preceded by a one-line header (line 1) that very briefly describes the harvested content and its length. Next to the one-line header, HTTP protocol response headers (from line 5 to line 16) are recorded, and then, multiple lines of HTML documents (from line 18 to line 43) are followed. We develop a WARC parser to separate the blocks into multiple individual HTML documents one by one and store them in different file folders according to what the domain extracted from the one-line header uniform resource locator (https://chato.cl/webspam/datasets/uk2007/contents/excerpt.txt) field the HTML document belongs to. Since the domain of each host is different, the folder name is the domain name.

3.3. Homepage Extraction and Check. A website contains at least one web page, and some websites are up to several hundred pages. The results obtained by users searching for keywords in search engines are just one web page for users, and it is challenging to get all the web pages of the website to which the current web page belongs. In other words, getting all the web pages is not easy, but getting the homepage is still relatively simple. Moreover, the homepage is the core of a website, covering the main content that a website wants to express to those who are visiting the website. For example, some companies, governments, and schools' websites will display related information about companies, governments, and schools such as history, main business, and contact information on the homepage. Statistics show that web spam pages are more inclined to improve their rankings in search engine results pages, especially homepages. Figure 3 shows that the percentage of a homepage with the largest PageRank value among all pages on the website of spam websites is higher than that of nonspam websites. It indicates that when spammers create a website, they intentionally make the homepage with the highest ranking. Some well-known algorithms for calculating page rankings include PageRank Page Score and TrustRank [32]. Therefore, it is very representative to check whether the homepage is spam. To some extent, the homepage can represent whether the entire website is spam.

The next step is to determine which HTML document is the homepage of a website. In the process of parsing HTML documents from WARC files, we have judged whether a web

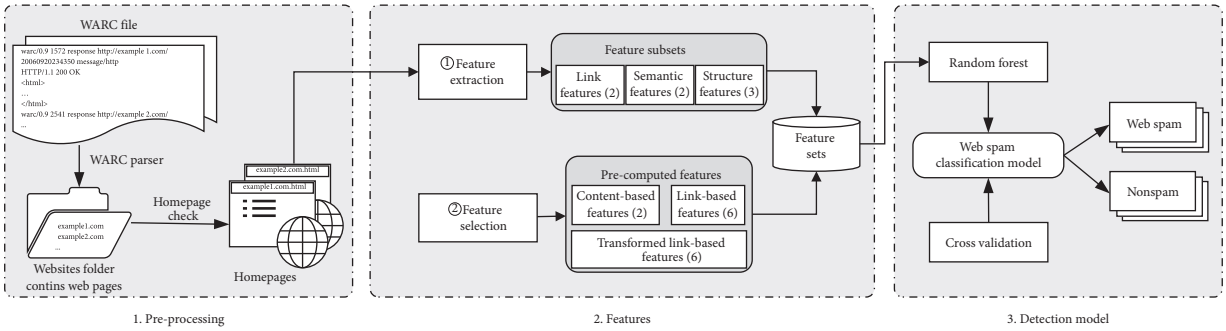


Figure 1: The framework of the proposed detecting web spam method.

```
1  warc/0.91572responsehttp://horwichrmicc.co.uk/20060920234350message/http uuid:e9004f91–fd17–4f40–b7d5–6c3e0b70f3d3
2  BUbiNG–guessed–charset: windows–1252
3  BUbiNG–content–digest: f05592c825fa9619306efcccd187391e
4
5  HTTP/1.1200 OK
6  x–powered–by: ASP.NET
7  connection: close
8  content–type: text/html
9  accept–ranges: bytes
10 content–location: http://horwichrmicc.co.uk/index.htm
11 server: Microsoft–IIS/6.0
12 content–length: 987
13 last–modified: Sun, 16 Apr2006 09:02:19 GMT
14 etag: “b5571d773461c61: b506b”
15 date: Wed, 26 Apr2006 13:11:10 GMT
16 ubi–http–equiv–charset: windows–1252
17
18 <html>
19 <head>
20 <meta http–equiv = “Content-Language”content = “en-us”>
21 <meta http–equiv = “Content-Type” content = “text/html; charset = windows–1252”>
22 <meta name = “horwich”“rmi”“winter hey lane”“pike”“reebok”“beehive""rivington”“lostock”“blackrod”“lee lane”“chorley
23 new road” content = “Microsoft FrontPage 5.0”>
24 <metaname = “ProgId” content = “FrontPage.Editor.Document”>
25 <title>Horwich RMI CC</title>
26 </head>
27 <body>
28 <palign = “center”>
29 <ahref = “home%20.htm”><img border = “0” src = “bannerentry.gif” width = “700” height = “400”></a>
30 </p>
31 <p align = “center”>
32 <b><script language = “JavaScript” type = “text/javascript” src = “http://pub24.bravenet.com/counter/code.php?id = 363671
33 &usernum = 2057536663&cpv = 2”>
34 </script><!-- End Bravenet.com Service Code--></b>
35 </p>
36 <p align = “center”>
37 <b><font size = “7”>Sponsored by Bespoke Flooring</font></b>
38 </p>
39 <palign = “center”> </p>
40 <p align = “center”>
41 <font color = “#FFFFFF”>THE OFFICIAL HORWICH R.M.I CRICKET WEBSITE</font>
42 </p>
43 </html>
```

Figure 2: An example data in the WARC format.

page is a homepage from the URL path roughly. We set every HTML document name as its pathname and store it under the website to which it belongs. Here are some simple rules, for example, the URL path is only the root path “/” could be as homepage, and the first-level path with the distinct keywords such as “index,” “home,” and “homepage,” is also the homepage. Of course, all web pages under some hosts do not match these rules, so a manual check is required.

3.4. Features. We extract some novel features from the source code of the web page and divide them into four categories mixed with existing features: homepage links features, semantic similarity features, homepage structure complexity features, and existing features. Although some features based on links, content, and structure have been used in previous papers, in this paper, we have studied these features from a different perspective.

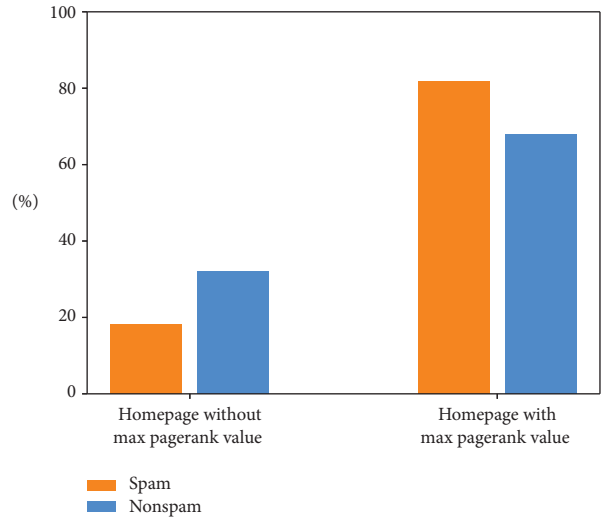


Figure 3: The percentage of homepage with max pagerank value of both spam and nonspam websites.

3.4.1. Link Features. It is necessary to consider link characteristics because the spammers deliberately set up a large number of hyperlinks between the spam websites to point to each other which can increase the clickthrough rate and the PageRank value of the website's homepage. There are some specific measures, such as inserting hyperlinks in the homepage to point to essential or well-known websites and attract other pages to point to their own pages or using information hiding technology to publish valuable information on the Internet, but hidden text or hyperlinks that are invisible to users, pointing to spam homepages. These practices all increase the entry link, also called indegree of the homepage of the spam websites. This makes the PageRank value of the homepage increase and leads to the advance of the spam page in the ranking of search engines, whereas the production of normal hosts is reasonable and standardized, and the host owner will not deliberately increase the homepage's incoming link. Many previous studies only focused on the number of all links, without considering external links and cross links separately. But, the impact of these two features on the construction of web spam is different. Based on this perspective, this paper extracts these two features separately. As discussed above, we propose two link features, number of external links and number of cross links, based on all the links in the homepage.

(1) Number of external links: external links defined as hyperlinks that point at an external domain which means any domain other than the domain the link exists on. We compared the domain of each link in the homepage with the domain to which the homepage belongs and counted the number of external links.

(2) Number of cross links: in contrast to external links, cross links also called internal links are links that, from within a website, point to another page which belongs to the same website. Similarly, the number of cross links is obtained by subtracting the number of external links from the total number of links.

3.4.2. Semantic Similarity Features. Generally, for content spam, the primary method is to repeat the same or similar keywords in large numbers. Some web pages directly copy the content of some standard high-quality websites. When users search for a specific keyword, these plagiarized websites will also have a relatively high ranking. Users will not be aware of this is spam page through only the restricted content displayed in the search engine results. These pages add partial anchor texts link to some marketing and authority websites, even malicious websites such as gambling and pornographic websites, to entice users to click and earn profits. This malicious behavior can be challenging to detect. Many web pages with interactive functions are easily used by spammers. For example, in the comments section of a blog, it is easy to evade censorship and spread malicious websites to entice users to click. In previous studies of content-based web spam, researchers mainly focused on the topics and keywords of the entire website. The method of detecting web spam from the entire content of the website is not accurate enough, and it will make web spam using this technology evade detection. Also, the partial web spam technique has not been widely studied yet. After analysis and manual check, we observe that the semantic analysis between the anchor text and the current web page is helpful for web spam detection. Therefore, we extract two semantic similarity features, namely, similarity of texts and links.

(1) Similarity of texts: the feature represents the semantic similarity between the textual description, also known as anchor text of the external links in the page and some textual description of the web page. It can reflect the similarity between the link's anchor text inserted in the web page and the main content of the web page.

(2) Similarity of links: the feature represents the semantic similarity of each domain of all links in the page and page's domain. It can reflect the similarity between the link inserted on this web page and the page's domain.

In this paper, we choose word mover's distance (WMD) as a metric to measure semantic similarity. The WMD is a novel distance function between text documents presented in [33]. In a supervised learning task, semantic similarities are useful for classification. WMD measures the difference between two texts and calculates the minimum distance that a word vector of one text "moves" to a word vector of another text. As shown in Figure 4, after removing stop words (not bold), the remaining words are embedded into a vector in the vector space. The WMD distance between the two short texts is the minimum cumulative distance calculated by word vectors in short text 1 travel to short text 2. Since the WMD distance can use the word-level semantic information represented by word2vec [34], it can achieve better results in the short text semantic distance calculation. Thus, WMD is suitable for computing the semantic similarity features in this paper. The smaller the WMD value, the more similar the two short texts. To automatically extract the semantic similarity features from the homepage's HTML

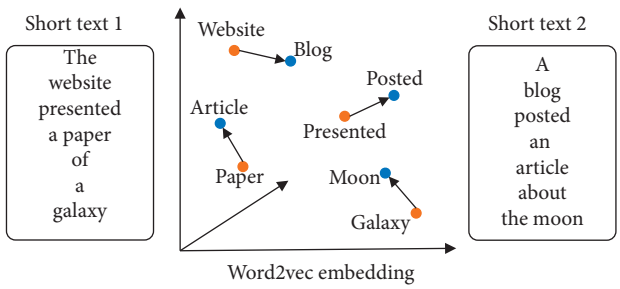


Figure 4: An illustration of the word mover's distance.

document, we propose an algorithm and complete it through the three steps: short text cleaning, represent words as vectors, and computing WMD distance. The pseudocode of this algorithm is shown in Algorithm 1.

As described in Algorithm 1, line 1 and line 2 show that the homepage and domain name of each website is required. Line 3 creates three lists: text_{hp} contains the homepage text, $\text{link}_{\text{external}}$ includes all external links, and $\text{text}_{\text{external}}$ is for storing the anchor text corresponding to each external link.

Line 4 to line 24 is the process of calculating semantic similarity features. There are five functions in the proposed method.

(i) **ExtractText()**: this function extracts the homepage's title, keywords, description in meta tags

(ii) **Collectlinks()**: this function extracts all external links of the homepage

(iii) **ExtractLinkText()**: it extracts the anchor text of each link in turn

(iv) **ExtractDomain()**: this function extracts the domain of each link in turn

(v) **WMD()**: it calculates the WMD distance between the homepage text and each anchor text of the external link and calculates the WMD distance between the homepage domain and each external link domain

Next, we introduce each step in detail.

Step 1 *Short Text Cleaning.* The HTML document of the homepage contains much information, but only a few parts are used to extract WMD features. To extract the title, the keywords and descriptions in the metafield, external links, and text description of every external link from the homepage of each website, we first need to parse the HTML tags with the Beautiful Soup Python library [35] and convert HTML entities to characters with the HTML Python library. Moreover, we remove some punctuations and stop words in raw texts. We utilize stop words from the NLTK library, which contains 127 English words. Besides, to ensure a hyperlink is an external link, it is necessary to extract each website's domain. Note that the external link includes neither relative paths nor links under the same domain which we mentioned in Section 3.1.1. Then, we splice the content of the homepage's title and keywords and description of metatag together as the text_{hp}. Besides, we should process two parts for each external link, the link itself and the anchor text of the link. Some websites contain more than one external link while some contain none. We push all the external links to a list as $\text{link}_{\text{external}}$. For each link's anchor text, we also push all the anchor texts corresponding to each link into the $\text{text}_{\text{external}}$ in turn, which is separated by spaces.

$$\begin{aligned}
\text{text}_{\text{hp}} &= \left\{\text{text}_{\text{title}}, \text{text}_{\text{keywords}}, \text{text}_{\text{description}}\right\}, \\
\text{link}_{\text{external}} &= \left\{\text{link}_1, \text{link}_2, \ldots, \text{link}_n\right\}, \quad n \in N^*, \\
\text{text}_{\text{external}} &= \left\{\text{text}_{\text{link}_1}, \text{text}_{\text{link}_2}, \ldots, \text{text}_{\text{link}_n}\right\}, \quad n \in N^*.
\end{aligned} \tag{1}$$

Step 2 *Represent Words as Vectors.* Since models accept numerical input only and the words in short texts are natural languages such as English, these words need to be converted into numerical forms or embedded in mathematical space. The vector mapped to real numbers is called word vectors. The embedding method is called word embedding, and word2vec is a kind of word embedding method. Word2Vec is a tool to transform the text processing into a vector in the multidimensional vector space, representing the text's semantic similarity based on the similarity in the vector space. We use a pretrained model, Google News [36], to train word vectors. It contains 3 million pretrained English word embeddings.

Step 3 *Computing WMD Distance.* After obtaining the original short texts and links' word vectors, we then calculate the WMD distance to represent the similarity, which is illustrated as follows:

$$\text{WMD} = \min_{T \geq 0} \sum_{i,j=1}^{n} T_{ij} c(i, j). \tag{2}$$

T is a sparse matrix where T_{ij} is the weight of word i in document d_i move to word j in document d_j, $c(i, j)$ is the Euclidean distance between word i and word j, as equation (3) shows, and x_i, x_j is the word i, j's word vector after embedding, respectively.

$$c(i, j) = \left\| x_i - x_j \right\|_2. \tag{3}$$

The sum of weight of word i to another document $\sum_{j=1}^{n} T_{ij}$ is equal to the weight of word i in the first document d_i, and the sum of weight of word j to another document $\sum_{j=1}^{n} T_{ij}$ is equal to the weight of word j in the first document d_j.

$$\begin{aligned}
\sum_{j=1}^{n} T_{ij} &= d_i, \quad \forall i \in \{1, \ldots, n\}, \\
\sum_{i=1}^{n} T_{ij} &= d_j, \quad \forall j \in \{1, \ldots, n\}.
\end{aligned} \tag{4}$$

d_i can be calculated by equation (5), where c_i is word i appear times in the document I.

$$d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}. \tag{5}$$

In the case of this article, document I corresponds to the text_{hp} and document J corresponds to the links $\text{link}_{\text{external}}$ and anchor text $\text{text}_{\text{external}}$ of external links. So, there are multiple short texts. We need to compare the semantic similarity of each external link with the homepage. Then, we calculate the average by equations (6) and (7), respectively.

$$\text{WMD}_{\text{text}} = \frac{\sum_{i=1}^{n} \text{WMD}\left(\text{text}_{\text{hp}}, \text{text}_{\text{link}_i}\right)}{n}, \tag{6}$$

$$\text{WMD}_{\text{link}} = \frac{\sum_{i=1}^{n} \text{WMD}\left(\text{domain}_{\text{hp}}, \text{domain}_{\text{link}_i}\right)}{n}. \tag{7}$$

3.4.3. Structure Complexity Features. It is necessary to consider the characteristics of the homepage's document object model (DOM) structure because we discover that many web pages use the same templates, such as domain parking services, personal blog websites, and some government websites. It can be considered that there are certain regularities. According to MDN [37], a web page is a document of which structure and content are represented as nodes and objects by DOM. Structural features can reflect the complexity of web pages. Similar web pages have a similar DOM structure because web spam is often a well-designed template, unlike normal web pages that have different characteristics. The previous method of identifying web spam mainly focused on the difference in content and links, but we also consider the web page's structural characteristics in this paper. Previous methods also studied the structure of HTML, such as extracting the number of <a> tags and <img> tags. However, this method may not be very effective for some specific websites, such as online shopping websites and stock picture websites, which have an obvious tendency of certain types of HTML tags. We not only analyze a certain type of tags but also analyze the complexity of the web page's structure from a more general perspective. We have analyzed the three features of the number and diversity of HTML tags and the depth of element nodes. For example, the domain parking service, where the parking platform often has a fixed template. We aim to identify such a web page. Thus, it is necessary to research the web page's structure, and we have extracted the following structural features from the source code of the homepage.

(1) Number of HTML tags: the DOM represents an HTML document as a tree structure of tags, which is a DOM tree. We only consider the element-type nodes, HTML tags. By traversing the DOM, the total number of tags contained in the homepage is calculated.

(2) Diversity of HTML tags: different types of websites have different distributions of their tags. For example, the web pages in some link factories contain a large number of hyperlinks, and the ¡a¿ tag implements these hyperlinks. There are many image tags in online stores, while personal blogs have many paragraph tags. We also counted each type of label on each homepage.

(3) Depth of element nodes: by traversing each branch of the DOM tree, we calculated the maximum depth of the DOM tree of the homepage. It reflects the complexity of a DOM tree structure and the complexity of the homepage's HTML structure.

3.4.4. Precomputed Features. There are 277 precomputed features in total, which categorize into 4 sets, direct features set with 2 features, content-based features set with 96 features, link-based features set with 41 features and transformed link-based features set with 138 features, which are obtained by mathematically transforming the link-based features. If all features are considered for experiments, it is evident that those with high dimensions will undoubtedly consume a lot of resources and cause a long execution time. In fact, many features are redundant. By removing several redundant features, both efficiency and accuracy can be improved. Therefore, we only take the features related to the homepage into consideration.

First of all, by checking the meaning of each feature, we preliminarily filter out 106 precomputed features from these four feature sets that are all about the homepage (hp), without considering the page with the maximum PageRank (mp) value. Then we select features from the 106 features. We use a new backward elimination approach, Smart-BT, proposed by Asdaghi and Soleimani [23] to accomplish feature selection. This method differs from sequence backward elimination in that it measures the impact on the classification result after eliminating a set of features, rather than eliminating a single feature. In summary, we extracted 7 new features and selected 14 features from the existing features. There are 21 features in total. These features will be input into the detection model.

3.5. Classification Algorithm. Judging whether a web page is spam or not is a problem with less clear boundaries. It is a subjective issue to some extent, so classifying web pages as spam or nonspam is challenging. Because of the apparent differences between spam and nonspam web pages in some features, we can use these features to build machine learning models that allow experts or researchers to identify web spam and reduce losses on the ground quickly. Classic algorithms, such as NB, logistic regression (LR), SVM, RF, convolutional neural networks (CNNs), RNN, and long short-term memory (LSTM), have different advantages and disadvantages. In [38, 39], a large-scale empirical comparison between these machine learning methods is presented. A CNN has an excellent performance spatial mapping, such as image data. An RNN is more suitable for sequence content, such as text analysis. But, an RNN has the problem of gradient disappearance; it is challenging to process long

Require:
hp: homepage of each domain
d_{hp}: homepage's domain
Initialize the list $text_{hp}$, $link_{external}$ and $text_{external}$ set to null
if (hp is not null) **then**
  $text_{hp}$: = **ExtractText** (hp)
  /* extract hp's title, keywords, description */
  $link_{external}$:= **Collectlinks** (hp)
  /* collect all external links in hp */
  if ($link_{external}$ and $text_{hp}$ is not null) **then**
    for each link $\in\in link_{external}$ **do**
      $text_{external}$: = **ExtractLinkText** (link)
      d_{link}: = **ExtractDomain** (link)
      /* extract link's anchor text */
    end for
    WMD_{text} = **WMD** ($text_{hp}$, $text_{external}$)
    /* computing the WMD distance between hp's text and external link's anchor text*/
    WMD_{link} = **WMD** (d_{hp}, d_{link})
    /* computing the WMD distance between hp's domain and external link's domain*/
  else if ($link_{external}$ is not null and $text_{hp}$ is null) **then**
    $WMD_{text} = 0$
  else
    $WMD_{text} = WMD_{link} = 0$
  end if
end if
return WMD_{text}, WMD_{link}

Algorithm 1: Calculating semantic similarity features from the homepage.

sequence data. LSTM can avoid the vanishing of gradient of conventional as a special case of the RNN.

Considering the characteristics of the dataset is imbalanced and features in the feature set are independent and based on the cost of different methods, the RF [40], which combines a multitude of decision trees, is more suitable for the problem solved in this paper. As an ensemble learning method for classification, RF solves the shortcomings of the weak generalization ability of decision trees since it predicts a sample by lots of decision trees voting for the final result. Furthermore, there are several advantages to selecting RF as the classifier. It is inherently easy to interpret and understand. Furthermore, RF algorithm is easy to implement and costs less than deep learning. Therefore, we chose to use RF as the automatic classifier in this paper.

4. Experiments and Evaluation

In this section, we have first illustrated the environment of experiments and detailed the source and composition of the dataset used in this paper. Then, the metrics utilized for the measurement of the performance of the proposed model are discussed, and later, experiment results are analyzed.

The experiments studies are conducted on the Ubuntu operating system. The homepages extraction and data preprocessing are developed in some libraries written in Python. Also, the process of model building and classification is implemented by scikit-learn [41] and keras [42] with TensorFlow backend [43]. The experimental environment configuration is shown in Table 1.

Since this paper focuses on the extraction and effects of novel features, the parameters in the detection model should be set or adjusted as little as possible. The simpler the machine learning model, the more likely it is that good experimental results are not based solely on specific samples. For example, in the detection model RF, we only set the number of trees parameter "n_estimators" to be 100 empirically. In fact, the default value of "n_estimators" changed from 10 to 100 in scikit-learn v0.22. There is no specific setting for the other hyperparameters, which means other hyperparameters are set by default. The advantage is that the classifier of this paper is not aimed at a specific dataset, but has generalization capabilities. There is a reason to believe that the proposed method is not too data dependent and easy to apply for new users. When encountering a new dataset, we only need to extract the features proposed in this paper according to the method in Section 3 and input these features into the classifier for classification. However, machine learning is dependent on data, and different data types have different targeting models, which we mentioned in Section 3.2. For data with similar regularities, the proposed method has generalization ability. Firstly, different web spam pages have similar characteristics, such as too many links for link-based web spam or a large number of repetitions of text content for content-based web spam. Secondly, cross-validation is used to evaluate the prediction performance of the model, especially the performance of the trained model on new data. The cross-validation can reduce overfitting to a certain extent and better evaluate the generalization quality of the model by repeatedly dividing the dataset.

Table 1: Experimental environment Configuration.

Designation	Information
Operating system	Ubuntu server 16.04 LTS
System configuration	CPU:Intel i7-7700 K, 3.60 GHz, 32 GB RAM GPU:Nvidia RTX 2080 8G
Python library	Genism 3.8.1 TensorFlow 2.0.0 Beautifulsoup4 4.4.1 NLTK 3.4.5 Keras 2.3.1 Scikit-learn 0.21.3 Matplotlib 3.1.1 Statsmodels 0.9.0
R library	pROC 1.16.2

4.1. Dataset. We run our experiments on the WEBSPAM-UK2007 dataset [31] which is a large collection of 105,896,555 pages in 114,529 hosts based on a crawl of the "uk" domain that was conducted in May 2007. It is also used as the Web Spam Challenge 2008 dataset. Although the amount of the dataset is large, only few were labeled by a group of volunteers. As shown in Table 2, among the all 6479 labeled data, we discard the data labeled as "undecided" because it means that the volunteers were still uncertain as to whether they were spam or nonspam and discard the data without content features. Meanwhile, we delete these data that their features are not complete. As a result, 5797 data remain.

As Table 2 depicts, the number of spam is 321, and the proportion of spam is only 6%; the ratio of samples of nonspam class to spam class is nearly 16 : 1, which means that the data are very imbalanced. In such scenarios, machine learning models cannot learn the characteristic behavior of the minority spam class. Classifying samples as spam or nonspam accurately presents considerable challenges. To address this issue, we re-extracted 1215 pieces of data after removing duplicate data in the original dataset from the results given by the top three [44] in the Web Spam Challenge 2008. These data were consistently labeled as "spam" by the top three teams. To a certain extent, these data can be considered reliable. Although we extracted more labeled data, these data were already 13 years old, so we also considered the newer dataset UK-2011 [45], which was derived from the WEBSPAM-UK2007 dataset. After deduplication, we find that all pages on many websites are invalid websites. "Invalid" means the source code of these pages has no content or is meaningless. There are probably the situations "301 Moved Permanently," "Object Moved," "This IP has been banned," and "302 Found." Since these pages have no research value, they need to be deleted. In addition, we have also removed some pages that are not in English. In the end, as Table 2 depicts, we have 6189 pages consisting of 4745 nonspam pages and 1444 spam pages.

We have a reason to believe that conducting the study with the 13-year-old dataset has certain limitations, such as whether it is suitable for today's rapid development of web applications. The meaningfulness of using the dataset is as follows.

Table 2: Statistics of experiments dataset.

Category	UK2007	UK2011	Dataset used in this paper
Nonspam	5476	1768	4745
Spam	321	1998	1444
Total	5797	3766	6189

(1) This dataset is a standard dataset in the field of web spam research, and its labels are judged by multiple scholars. It can be considered that its labels are authoritative.

(2) Although it has passed 13 years, it is still in the web 2.0 era now. Developers build web pages, especially web spam, with some commonality technologies before and nowadays, which indicates the dataset is universal. Moreover, during our manual check process, we find that some websites are still active.

4.2. Experimental Design. We conduct three sets of experimental studies to evaluate the performance of our model fully: (1) we first evaluate the performance of our proposed method and verify the effectiveness of the novel feature; (2) we also compare the performance of our method with some popular web spam detection systems; and (3) we use hypothesis testing to verify the validity of our method and analyze the importance of features.

4.2.1. Experiment for Performance of Proposed Model. We first examine the performance of the RF model on the dataset and compare the results with benchmark models. Considering the dataset is inadequate, especially the data for the minority class and different samples or different partitions of the dataset may cause the result to be optimistically biased, we take cross validation to train our dataset. As a potent tool in machine learning and deep learning, cross validation can ensure that every page in the dataset can be used in the experiment process. In this way, we ensure the full use of the data and manage to make the experimental results less biased. Thus, we apply 5-fold cross validation to train our dataset in all detection models. We input all the features that comprise existing features and novel features into classification approaches to determine whether a page is spam or not. We have also investigated some benchmark traditional machine learning algorithms such as NB, LR, and SVM and some benchmark deep learning algorithms including CNN, RNN, and LSTM as basic comparative experiments. Secondly, we compare the classification effects of each model on only existing features and all features.

4.2.2. Experiment for Comparison of Detection Rate. We also have compared some state-of-art methods; for example, Mittal and Juneja [46] presented a mutual information-based feature selection method which selects content-based features and with a SVM classifier to distinguish web spam, Makkar et al. [26] used PCA and RFE to deal with link-based features and incorporated the features into an RNN classifier to detect spam, and Asdaghi and Soleimani [23] proposed an

effective feature selection method to select fewer features and put the features into NB model to achieve high performance. We used the same method as these papers to divide the dataset into training, validation, and test sets. In order to make a comparison on the dataset used in this paper, we reproduced these experiments.

4.2.3. Experiment for Validity of the Proposed Features. As one of the models considered is the LR model and the best model is RF, followed by comes LR, we believe that reporting the logistic regression model results with statistical inference would be very useful for more than one reason. We use statsmodels [47] for the estimation of many different statistical models. Firstly, it would verify that some of the features identified in Section 3.1 can actually be used to detect spam, and it would demonstrate which variables are the most important in this regard. Secondly, it could then be used as a benchmark against which the other models could be compared. We use the open-source package "pROC" provided in Robin et al. [48] to compare two different models' area under the curve (AUC). It helps us compare the superiority of different models more rigorously, especially when the p value is less than 0.05, and the results are more convincing.

We also use two methods including mean decrease impurity (MDI) and mean decrease accuracy (MDA) also known as permutation importance (PI) in RF for feature importance analysis. There are two main problems of impurity-based feature importance methods are that it biased towards high cardinality features, and the impurity-based importances are computed on the training set. Hence, it is not certain that features are also useful on the test set, whereas MDA is an alternative that can mitigate those limitations. We add up the ranked results of each feature and calculate the average importance score because of cross validation.

4.3. Evaluation Metrics. In binary classification problems, the most popular performance evaluation indicators are accuracy, precision, recall, and F1 score, which are described in detail as follows.

Accuracy is the number of correct predictions over the number of total predictions of the model.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{8}$$

We judge a sample with a prediction score greater than the threshold as positive (spam class), and the threshold is 0.5. Where true positive (TP) is the number of spam samples that are correctly classified, true negative (TN) is the number of nonspam samples that are correctly identified, false negative (FN) is the number of spam samples that are mistakenly classified as a nonspam sample, and false positive (FP) is the number of nonspam samples that are mistakenly classified as spam.

Recall, also called true positive rate (TPR), is the proportion of the number of spam samples was identified correctly as spam in all spam samples and defined in equation (9). Precision is the proportion of the true predictions of the spam samples over total samples predicted as spam which is defined in equation (10).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{9}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{10}$$

F1 score is the harmonic average of precision and recall and defined in the following equation:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{11}$$

False positive rate (FPR) is defined as follows:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{12}$$

Receiver operating characteristics (ROC) curve must be regarded as a widely used indicator when it comes to metrics for performance evaluation in classification problems. Because the ROC curve has an outstanding characteristic, the ROC curve can remain unchanged when the distribution of positive and negative samples in the test set changes. Also, AUC, which utilizes TPR and FPR (equations (9) and (12), respectively), represents classifiers' performance. The larger the AUC value, the better the classifier is in detecting web spam.

4.4. Results and Analysis

4.4.1. Classifier Performance Result. We show the performance of the binary classifiers in detecting web spam in Table 3, which demonstrates the results of different baseline models using the evaluation indicators described in Section 4.3. Figure 5 illustrates the graphical view of the performance of the different classifiers in web spam detection. It can be seen from the table and figure that the RF model yields the best performance in all aspects of our experiments, with a precision rate of 0.929 and a recall rate of 0.930 and has the largest curve area. Trees of RF algorithm are independent during the training process. The final result is obtained by voting of all trees. For imbalanced data sets, RF can balance errors [49]. LR is closely followed, and SVM, CNN, and RNN perform well but relatively worse. However, NB and LSTM performance are slightly worse. NB is relatively simple, more sensitive to minority class data. LSTM is suitable for longer sequence data, so the dataset in this paper does not highlight its advantages. The result of experiment for performance of the proposed model demonstrates that the RF model can use the features effectively. We can conclude that the selected novel features combined with the chosen classifier RF yields better results. In Table 4, we can see the existing features and novel features, as well as the results of all features under different evaluation indicators. Without novel features, the result is inferior to that of with novel features, which means that the novel features we extracted are practical.

Table 3: Results of different classification models. These bold values indicate the best performance under different evaluation metrics.

Model	Accuracy	Precision	Recall	F1 score
NB	0.850	0.843	0.850	0.844
LR	0.888	0.884	0.888	0.883
SVM	0.891	0.899	0.891	0.880
RF	**0.930**	**0.929**	**0.930**	**0.929**
LSTM	0.810	0.835	0.810	0.758
CNN	0.859	0.861	0.859	0.842
RNN	0.875	0.872	0.875	0.865

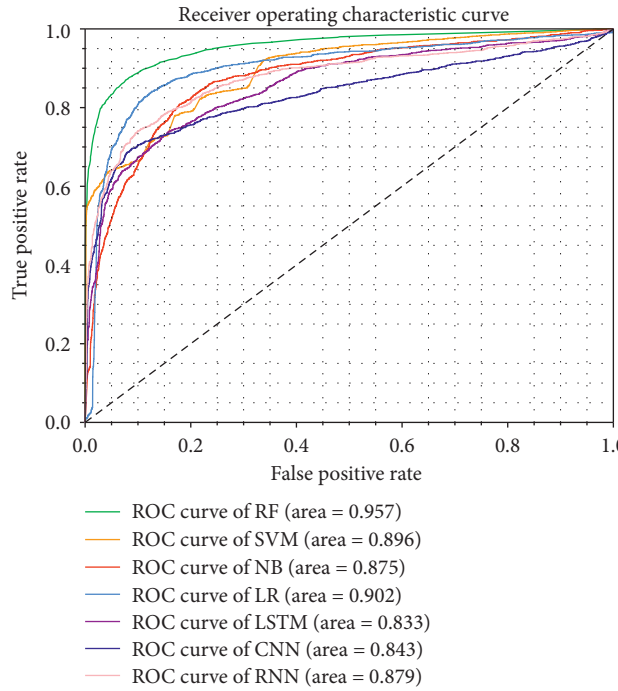


Figure 5: The ROC curve of 7 models (5-fold cross validation).

Table 4: Results of using different 3 feature sets on random forest model.

Feature set	Accuracy	Precision	Recall	F1 score	AUC
Selected existing features	0.911	0.909	0.911	0.909	0.937
Novel features	0.911	0.910	0.911	0.907	0.917
Selected existing + novel features	0.930	0.929	0.930	0.829	0.957

4.4.2. Comparison Experiment Result. In this paper, we reproduced three representative methods as comparative experiments that we explained in Section 4.2. The three studies were chosen for the comparisons based on the consideration that the researches were relatively new and their results performed well. Also, they all used the same dataset as this paper. As in Table 5, which demonstrates the results of three state-of-art methods, paper [46] achieved an F1 score of 0.883 on our dataset, which was less than our method by nearly 5%. Our method uses fewer features and achieves better results. It is concluded that these methods are worse than our method and our proposed method performs well.

4.4.3. Validity Verification Result. Table 6 demonstrates some regression results of the features used in this paper. It includes each feature's coefficients, standard error, and p value by logit regression analysis. It can be seen that the novel features contribute significantly to the model.

"Two ROC curves are 'paired' if they derive from multiple measurements on the same sample" described by Robin et al. [48]. Thus, we compare the ROC curve of RF with other models' ROC curves, respectively. We use "roc.test()" command from "pROC" package in R to calculate the p value. All paired ROC curves' p value is less than 2.2e-16, which is far less than 0.05. We could say that the RF model (AUC = 0.957) has an AUC that is significantly

Table 5: Comparison of the results using different classification methods.

Method	Number of features	Accuracy	Precision	Recall	F1 score
Our method	21	**0.930**	**0.929**	**0.930**	**0.929**
The method of Mittal et al. [46]	70	0.890	0.888	0.890	0.883
The method of Makkar et al. [26]	41	0.872	0.869	0.872	0.870
The method of Asdaghi et al. [23]	28	0.863	0.869	0.863	0.865

Table 6: Logit regression results.

Type	Feature	Coefficients	Std. error	Pr(>—z—)
Selected existing features	HST_19	0.7254	0.338	0.032
	HST_20	2.3106	0.420	0.000
	outdegree_hp	0.0066	0.003	0.015
	pagerank_hp	−9.712e + 04	4.2e + 05	0.817
	truncatedpagerank_1_hp	−1.043e + 05	1.35e + 06	0.939
	truncatedpagerank_2_hp	−1.079e + 05	2.35e + 06	0.963
	truncatedpagerank_3_hp	−1.101e + 05	4.28e + 06	0.980
	truncatedpagerank_4_hp	−1.084e + 05	2.74e + 06	0.968
	L_outdegree_hp	−0.0425	0.002	0.000
	L_pagerank_hp	0.0472	0.012	0.000
	L_truncatedpagerank_1_hp	6.4183	0.665	0.000
	L_truncatedpagerank_2_hp	−8.9288	1.766	0.000
	L_truncatedpagerank_3_hp	3.5543	2.380	0.135
	L_truncatedpagerank_4_hp	−0.9848	1.369	0.472
Novel features	Diversity of HTML tags	−0.0523	0.011	0.000
	Depth of element nodes	−0.0123	0.002	0.000
	Number of HTML tags	−0.043	0.000	0.000
	Number of external links	0.0460	0.003	0.000
	Number of cross links	0.0173	0.002	0.000
	Similarity of texts	−0.0816	0.029	0.004
	Similarity of links	1.7910	0.249	0.000

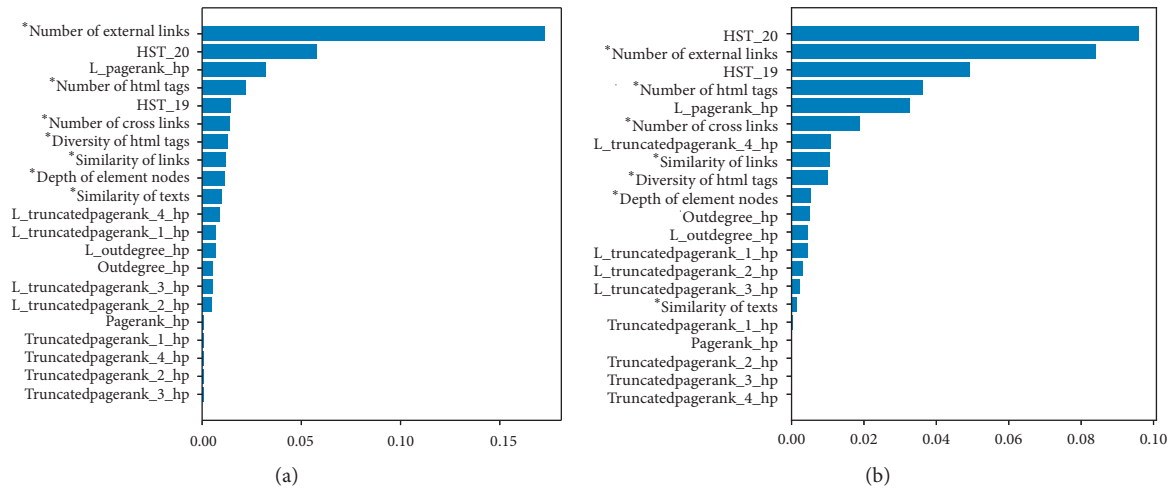


Figure 6: Random forest feature importance. (a) Mean decrease impurity. (b) Mean decrease accuracy.

greater than the second-best model (AUC = 0.902). Also, the results are not accidental which proves that our method is correct and effective.

As can be observed in Figure 6, the Figure 6(a) is the result of using MDI, and the Figure 6(b) is the result of using MDA. The results of the two RF feature importance ranking methods are not exactly the same. The ones with * on the *Y*-axis are novel features. It is clear that the novel features extracted in this paper rank top overall. The advantage of the features proposed in this paper is that it is convenient to extract, whereas the precomputed existing features extraction requires more stringent conditions such as construction of web graph. The novel features are general and easily accessible.

5. Conclusions and Discussion

Based on current research, this paper proposes a new method to distinguish web spam. We introduce a set of novel features about the homepage which we manually checked. In the meantime, we use the feature selection algorithm Smart-BT [23] to reduce the precomputed existing features' dimension so that the method's computational cost will decrease. Then, we use the RF model to discriminate against web spam with efficient identification. The experiment results showed that this method could reach a state-of-art level compared with other methods. Besides, the model with novel features which are are impressive to web spam detection is more superior and valid than the model with only existing features. Since this paper takes homepage only into account, the method is general and extensible because obtaining all pages of a website is not easy in most times. We acknowledge that some of the biases of our dataset might affect the result. Our method may not work well as the web spam evolves because the boundary between spam and nonspam is likely to blur. Also, we only analyzed statically from the source code without considering the dynamic parts such as JavaScript code, so our method has limitations for web spam that uses dynamic technology. For example, cloaking and redirection web spam. The proposed method only focuses on the homepage of a certain website without confirming whether the website returns different content for users and search engines so that there is a certain error in detecting this type of web spam. Moreover, many malicious websites redirect to other pages to improve rankings. There are many ways to achieve redirection, such as the redirection field in the meta tag and dynamic scripts in JavaScript. The proposed method does not pay attention to the JavaScript code and redirection web spam detection is not accurate enough.

In the future, mining more efficient features based on static and dynamic analysis and using a classifier with the ability of high accuracy would be an interesting direction. This will be the direction we will consider later.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant no.61902265, Sichuan Science and Technology Program under Grant nos.2020YFG0047 and 2020YFG0076, and the Fundamental Research Funds for the Central Universities.

References

[1] Search engine marketing statistics 2020. https://www.smartinsights.com/search-engine-marketing/search-engine-statistics/.

[2] Google search statistics. https://www.internetlivestats.com/google-search-statistics/(2019).

[3] Google organic click-through rates in 2014, https://moz.com/blog/google-organic-click-through-rates-in-2014(2019).

[4] How far down the search engine results page will most people go? https://www.theleverageway.com/blog/how-far-down-the-search-engine-results-page-will-most-people-go/(2019).

[5] Z. Gyongyi and H. Garcia-Molina, "Web spam taxonomy," in *Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web*, Chiba, Japan, May 2005.

[6] A. Lina, "Towards evaluating web spam threats and countermeasures," *International Journal of Advanced Computer Ence and Applications*, vol. 9, no. 10, 2018.

[7] M. Najork, *Web Spam Detection*, pp. 1–5, Springer, New York, NY, USA, 2017.

[8] M. Mahmoudi, A. Yari, and S. Khadivi, "Web spam detection based on discriminative content and link features," in *Proceedings of the 2010 5th International Symposium on Telecommunications*, Tehran, Iran, December 2011.

[9] R. K. Roul, S. R. Asthana, M. Shah, and D. Parikh, "Detecting spam web pages using content and link-based techniques," *Sadhana*, vol. 41, no. 2, pp. 193–202, 2016.

[10] J. Deng, H. Chen, and J. Sun, "Uncovering cloaking web pages with hybrid detection approaches," in *Proceedings of the 2013 International Symposium on Computational and Business Intelligence*, pp. 291–296, IEEE, New Delhi, India, August 2013.

[11] R. Duan, W. Wang, and W. Lee, "Cloaker catcher: a client-based cloaking detection system," 2017, https://arxiv.org/abs/1710.01387.

[12] Y. Liu, F. Chen, W. Kong et al., "Identifying web spam with the wisdom of the crowds," *ACM Transactions on the Web*, vol. 6, no. 1, pp. 1–30, 2012.

[13] J. Piskorski, M. Sydow, and D. Weiss, "Exploring linguistic features for web spam detection: a preliminary study," in *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web*, pp. 25–28, Beijing, China, 2008.

[14] A. Benczúr, I. Bíró, K. . Csalogány, and T. Sarlós, "Web spam detection via commercial intent analysis," in *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web*, pp. 89–92, Banff, Canada, May 2007.

[15] I. Bíró, D. . Siklósi, J. . Szabó, and A. A. Benczúr, "Linked latent dirichlet allocation in web spam filtering," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pp. 37–40, Madrid, Spain, April 2009.

[16] Y. Liu, R. Cen, M. Zhang, S. Ma, and L. Ru, "Identifying web spam with user behavior analysis," in *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web*, pp. 9–16, Beijing, China, April 2008.

[17] I. Luca, T. Kurt, A. Kapravelos, O. Comanescu, J.-M. Picod, and E. Bursztein, "Cloak of visibility: detecting when machines browse a different web," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP)*, pp. 743–758, IEEE, San Jose, CA, USA, May 2016.

[18] G. Liu, X. Huang, X. Liu, and H. Fan, "Document sentiment modeling based on topic attention hierarchy memory

network," *Journal of Sichuan University. Natural Science Edition*, vol. 56, no. 5, pp. 55–64, 2019.

[19] S. H. Reza Mohammadi and M. A. Zare Chahooki, "Web spam detection using multiple kernels in twin support vector machine," 2016, https://arxiv.org/abs/1605.02917.

[20] J. Fdez-Glez, D. Ruano-Ordás, R. Laza, J. R. Méndez, P. Reyes, and F. Fdez-Riverola, "WSF2: a novel framework for filtering web spam," *Scientific Programming*, vol. 2016, Article ID 6091385, , 2016.

[21] Y. Mei, J. Zhang, J. Wang, J. Gao, T. Xu, and R. Yu, "The research of spam web page detection method based on web page differentiation and concrete cluster centers," in *Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications*, pp. 820–826, Springer, Tianjin, China, June 2018.

[22] H. Jelodar, Y. Wang, C. Yuan, and X. Jiang, "A systematic framework to discover pattern for web spam classification," in *Proceedings of the 2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp. 32–39, IEEE, Vancouver, Canada, November 2017.

[23] F. Asdaghi and A. Soleimani, "An effective feature selection method for web spam detection," *Knowledge-Based Systems*, vol. 166, pp. 198–206, 2019.

[24] M. S. Renato, T. A. Almeida, and A. Yamakami, "Artificial neural networks for content-based web spam detection," in *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, July 2012.

[25] Y. Li, X. Nie, and R. Huang, "Web spam classification method based on deep belief networks," *Expert Systems with Applications*, vol. 96, pp. 261–270, 2018.

[26] A. Makkar, M. S. Obaidat, and N. Kumar, "FS2RNN: feature selection scheme for web spam detection using recurrent neural networks," in *Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, Abu Dhabi, UAE, December 2018.

[27] A. Belahcen, M. Bianchini, and S. Franco, "Web spam detection using transductive (inductive graph neural networks," in *Advances in Neural Networks: Computational and Theoretical Issues*, pp. 83–91, Springer, Berlin, Germany, 2015.

[28] H. Fu, X. Xie, Y. Rui, N. Z. Gong, G. Sun, and E. Chen, "Robust spammer detection in microblogs," *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 6, pp. 1–31, 2017.

[29] N.'A. Maulat Samsudin, C. F. Mohd Foozy, N. Alias, P. Shamala, N. F. Othman, and W. I. Sofiah Wan Din, "Youtube spam detection framework using naïve bayes and logistic regression," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 3, pp. 1508–1517, 2019.

[30] E. Ezpeleta, M. Iturbe, I. Garitano, I. Velez de Mendizabal, U. Zurutuza, and António Sáez, Edited by A. Enrique, de la Cal, Á. Herrero, and H. Quintián, Eds., "A mood analysis on youtube comments and a method for improved social spam detection," in *Hybrid Artificial Intelligent Systems*, E. Corchado, Ed., pp. 514–525, Springer International Publishing, Cham, Switzerland, 2018.

[31] Laboratory of Web Algorithmics (http://law.di.unimi.it/)s University of Milan. Web collection uk-2006/uk-2007. https://github.com/keras-team/keras (2019).

[32] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen, "Combating web spam with trustrank," in *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB)*, Sunnyvale, CA, USA, November 2004.

[33] M. Kusner, Yu Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *Proceedings of the International Conference on Machine Learning*, pp. 957–966, Lille, France, July 2015.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, https://arxiv.org/abs/1301.3781.

[35] Beautiful soup documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc (2020).

[36] Googlenews-vectors-negative300. https://code.google.com/archive/p/word2vec/(2019).

[37] Document object model-introduction to the dom. https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model (2020).

[38] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 161–168, New York, NY, USA, June 2006.

[39] K. L. Goh and A. K. Singh, "Comprehensive literature review on machine learning structures for web spam classification," *Procedia Computer Science*, vol. 70, pp. 434–441, 2015.

[40] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] Keras: Deep learning for humans. https://github.com/keras-team/keras (2019).

[43] M. Abadi, A. Agarwal, B. Paul et al., "Tensorflow: large-scale machine learning on heterogeneous distributed systems," 2016, https://arxiv.org/abs/1603.04467.

[44] Web spam challenge phase iii results. http://webspam.lip6.fr/wiki/pmwiki.php?n=Main.PhaseIIIResults (2019).

[45] H. Wahsheh, I. Abu Doush, M. Al-Kabi, I. Alsmadi, and E. Al-Shawakfa, "Using machine learning algorithms to detect content-based Arabic web spam," *Journal of Information Assurance & Security*, vol. 7, no. 1, 2012.

[46] S. Mittal and A. Juneja, "Feature selection-model-based content analysis for combating web spam," *Computer Science & Information Technology*, vol. 6, pp. 27–34, 2016.

[47] S. Seabold and J. Perktold, "Statsmodels: econometric and statistical modeling with python," in *Proceedings of the 9th Python in Science Conference*, Austin, TX, USA, July 2010.

[48] X. Robin, N. Turck, H. Alexandre et al., "proc: an open-source package for r and s+ to analyze and compare roc curves," *BMC Bioinformatics*, vol. 12, no. 1, pp. 1–8, 2011.

[49] C. Chen, A. Liaw, L. Breiman et al., *Using random forest to learn imbalanced data*, p. 24, University of California, Berkeley, CA, USA, 2004.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8881760, 19 pages
https://doi.org/10.1155/2020/8881760

Research Article

Binary File's Visualization and Entropy Features Analysis Combined with Multiple Deep Learning Networks for Malware Classification

Hui Guo,[1] **Shuguang Huang**,[1] **Cheng Huang**,[2] **Fan Shi**,[1] **Min Zhang**,[1] **and Zulie Pan**[1]

[1]*College of Electronic Engineering, National University of Defense Technology, Hefei 230011, China*
[2]*College of Cybersecurity, Sichuan University, Chengdu 610065, China*

Correspondence should be addressed to Fan Shi; shifan17@nudt.edu.cn

Received 2 September 2020; Revised 6 November 2020; Accepted 20 November 2020; Published 4 December 2020

Academic Editor: Liguo Zhang

In recent years, the research on malware variant classification has attracted much more attention. However, there are still many challenges, including the low accuracy of classification of samples of similar malware families, high time, and resource consumption. This paper proposes a new method of malware classification based on multiple visual features of malware and deep learning algorithms. In prior research, visualization techniques and entropy demonstrated exemplary performance in many areas. This paper extracts numerous visual features from the raw bytes and entropy sequence of the malware, which makes it more sensitive to malware samples of similar families and endows it the ability to classify malware variants more accurately. To evaluate the proposed method, this paper conducted a series of experiments on two malware datasets with a total of more than 20,000 samples provided by the Malware Research Lab and Microsoft Research. Through experiments, the method showed its superiority compared with some leading malware visual classification methods, achieving good performance on the accuracy with at least 1% improvement. The accuracy of the method even could reach 99.73% and 99.54%, respectively, on the two datasets.

1. Introduction

In recent years, the exponential growth of malware has posed a serious threat to cyber security. According to the Symantec internet security threat report [1], 246,002,762 new malware variants were monitored in 2018, and the number of new malware variants has exceeded a billion in the past three years. Hackers are increasingly inclined to use techniques such as packing and encryption to slightly modify the original malicious code to create new malware variants. Therefore, the rapid and accurate identification of malware variants can effectively assist the cyber security personnel in grasping their harmfulness and other attributes, which has significant research value.

With the development of machine learning techniques, data mining methods are often used to analyze malware, and many features-based detection methods are proposed [2]. These methods first extract the features of malware and then detect malware by using these features. This approach has become the mainstream method of malware detection. Currently, malware detection methods consist primarily of two types of approaches: static-feature-based detection and dynamic-features-based detection. The malware detection based on static features mainly analyzes the raw bytes of malware or disassembles the malware to analyze its opcodes, file structure, and other file attributes. The dynamic detection methods often extract behavioral features such as API operation sequence, file operations, and network communication during its process by running malware samples in a virtual environment or sandbox.

1.1. Need for This Study. However, code obfuscation technology could modify malware, increase the difficulty of code reverse, and reduce the performance of static malware detection. Dynamic detection is more robust, but it is still disturbed by different kinds of countermeasures (e.g., strict triggering conditions for malicious behavior or increasing

waiting time to avoid dynamic detection), reducing the detection performance. Moreover, the time and resources consumption of malware execution is often expensive.

In recent years, with the rapid development of deep learning algorithms and image recognition technologies, rather than focusing on these nonvisual detection methods, many researchers have proposed new classification methods based on malware visualization methods [3–7]. They transform the raw bytes of malware into grayscale images and extract the malware texture features for classification. Besides, these visualization methods are also proven to be more robust than the static method [3] and provided performance roughly equivalent to dynamic detection methods, but in less time [5].

Although the visualization detection methods can handle some code obfuscation problems and show some superiority, they still have some limitations and challenges. The malware variant has many homologous parts with its ancestors, which makes it have a strong correlation in visual characteristics. However, the malware samples of similar families also have some correlations, showing some similarities in the visual features, which would cause some interference to the malware classifier and reduce the performance. The latest related methods, combined with deep learning algorithms, have partially alleviated the problem. But it is still not entirely resolved. Therefore, a critical challenge in the field is how to effectively distinguish the samples of similar malware families and improve classification performance.

1.2. Major Contributions of the Study. To address the above challenges, this paper proposes a new malware classification method based on malware visual features, entropy features, and deep learning algorithms. In the field of image recognition, the deep learning algorithm often performs better than traditional feature extraction algorithms. It can enable computers to automatically learn important features in images to improve the performance. Besides, the entropy, as a measure of randomness or uncertainty, has achieved good performance in detecting encrypted, compressed malware [8–11]. Therefore, the entropy feature and deep learning algorithms may assist us in learning more detailed features contained in malware samples to classify the malicious code more accurately and improve the classification performance of malware samples of similar families. Based on this hypothesis, this paper extracts multiple features of the raw bytes and entropy features of malware based on the deep learning algorithms.

First, we transform the raw bytes of malware into RGB images in bytes sequence order and extract bottleneck features based on the convolutional layer of a deep learning network. Then, the malware is converted into another RGB images using a visualization method containing location information, and the bottleneck feature is also extracted. Besides, this paper extracts the entropy sequence of the malware and then extracts the visualization features of the entropy sequence based on the grayscale image visualization method. At last, a machine learning malware classifier is trained based on these three types of features.

To evaluate the proposed method, this paper conducted cross-validation experiments on two different malware datasets with a total of more than 20,000 samples provided by the Malware Research Lab and Microsoft Research. The experimental results show that the proposed method could significantly improve the classification performance of malware samples of similar families and achieve better performance on the accuracy with at least 1% and 2% improvement on two datasets compared with the leading malware visual classification methods, and the accuracy of the method even could reach 99.73% and 99.54%, respectively, on the two datasets.

Overall, we make the following contributions:

A new malware classification method was proposed based on the raw bytes of malware. It transforms malware into three kinds of images based on multiple visualization methods, including visualization methods containing location information and entropy features. The combination of these three types of features could effectively improve the performance of malware classification.

This paper introduces a visualization method of the entropy sequence. Unlike other traditional methods, we mine entropy features based on deep learning algorithms, and the experimental results show that it can play an essential role in malware classification.

Extensive experimental results showed that the proposed method almost solved the problem of confusion of similar family samples in malware classification and effectively improved the performance of malware classification, which is of great significance for the future research.

2. Related Work

In this section, the related research of malware classification is presented, mainly including malware detection methods based on static features, dynamic features, and visualization features.

2.1. Malware Detection Based on Static Features. The static feature of malware includes raw bytes feature, opcode, PE header features, import and export table, CPU register features, and static API sequence. The malware static detection method extracts these features to train a classifier for malware classification. Piyanuntcharatsr et al. [12] processed the byte sequence and opcodes with the N-gram algorithm and trained a malware classifier based on the decision tree algorithm. Feng et al. [13] extracted the bytes features of the malware and trained the malware classifier based on the support vector machine (SVM) algorithm. Raff and Nicholas [14] processed the raw bytes of malware based on the k-nearest-neighbor (KNN) algorithm to train a malware classifier. Kong and Yan [15] disassembled the malware samples and extracted the function call graphs for malware classification. Upchurch and Zhou [16] extracted the raw bytes and disassembly features of malware and classified the

malware based on multiple methods. The static detection method does not need to run malicious code files and would not cause harm to the system. However, it relies heavily on the file features of malware, which is less robust and vulnerable to code obfuscation technologies.

2.2. Malware Detection Based on Dynamic Features. Aiming at the problem of low robustness of static features, researchers have executed malware samples in virtual machines, sandboxes, and other virtual environments. They extract the malware behavioral features, including file operation behaviors, network communication behaviors, and runtime API call sequences, and train malware classifiers based on machine learning algorithms. Kawaguchi and Omote [17] extracted the dynamic API features of malware samples and classified the malware based on the features of the functions and machine learning algorithms. Vadrevu and Perdisci [18] proposed a novel malware testing framework and executed the malware samples in an analysis environment and extracted the network activity for malware classification. The experimental results in the paper showed that using their framework, they could reduce the malware execution time. Kim et al. [19], Dai et al. [20], and Lin et al. [21] also proposed malware classification methods based on the behavior features of malware samples. Malware detection technology based on dynamic features overcomes the impact of code obfuscation technology and has stronger robustness. However, these methods are still challenged by different kinds of countermeasures [5] (e.g., set waiting time measures and operation environment detection measures). Moreover, dynamic detection is time-consuming because it usually consumes time to execute malware samples, making it unsuitable for large datasets.

2.3. Malware Detection Based on Visualization Features. With the rapid development of image recognition technology, many scholars have utilized visualization technology for malware detection. Nataraj et al. [3] proposed a new method to extract malware visualization features and classify malware based on the k-nearest-neighbor (KNN) algorithm. They transformed the malware samples into grayscale images and extracted the texture features based on the GIST algorithm. Compared with the traditional feature-based methods, their method could provide more improved results [5]. Kosmidis and Kalloniatis [22] conducted more research on the visualization feature of malware and evaluated the performance of different machine learning algorithms. Inspired by these studies, Naeem et al. [6, 23], Xiaofang et al. [24], and Hashemi and Hamzeh [25] proposed many new malware classification methods based on different visualization features of malware images (e.g., DSIFT, LBP, and SURF). Moreover, the researchers also combined visualization technology with other malware characteristics for malware detection. Zhang et al. [26] visualized the opcode sequences of malware and achieved good accuracy for a small training set. Han et al. [27] visualized the dynamic features and opcode sequences of malware and extracted visualization features for malware classification.

Besides, with the rapid development of deep learning technology, Cui et al. [4], Rezende et al. [28], and Tang et al. [7] proposed better methods of malware classification based on deep learning algorithms. Cui et al. [4] visualized the malware samples and classified the malware based on convolutional neural networks. They also propose a method to improve the classification performance in the case of insufficient training samples. Tang et al. [7] also proposed a malware classification method based on deep learning algorithms. They significantly alleviated the lacking data problem and improved the performance of malware classification. Rezende et al. [28] proposed a malware classification method based on transfer learning algorithms. They extracted the visualization features of the malware based on the VGG16 network and achieved good performance. The visualization-feature-based detection methods are efficient and provide better performance relative to some traditional methods, but they still have some challenges. For example, the classification accuracy of malware samples of similar families is relatively poor. The latest related methods, combined with deep learning algorithms, have partially alleviated the problem. But it is still not entirely resolved, and the accuracy of classification needs to be further improved.

2.4. Malware Detection Based on Entropy Features. Besides, many scholars have detected the malware based on the entropy features. Wojnowicz et al. [8], Bat-Erdene et al. [9], and Liu et al. [10] proposed different kinds of methods for malware classification based on the entropy features of malware. Wojnowicz et al. [8] proposed a method for the detection of parasitic malware based on the entropy features and achieved good performance. Bat-Erdene et al. [9] proposed a method to detect the packing algorithm of malware based on the entropy features. Liu et al. [10] extracted the entropy sequence of malicious documents and detected the malware based on the machine learning algorithms. Moreover, Canfora et al. [11] extracted the entropy features of Android malware and proposed a malware detection method. Therefore, the entropy features of malware should also be one of the critical attributes, which may play an essential role in the research of malware classification.

3. Methods

3.1. Overview. First, we extract the raw bytes of malware to mine features hidden in the malware. The raw bytes of the malware are processed in two ways (i.e., extracting raw byte stream data and extracting entropy sequence data). The extracted data are processed based on three visualization methods to obtain different kinds of visualization representations. Then, three types of features (byte sequence level RGB feature, location information level RGB feature, and entropy sequence feature) are extracted based on the transfer learning algorithms. At last, a malware classifier is trained based on these features, and it would classify new malware samples to identify their families. An overview of our method is shown in Figure 1.

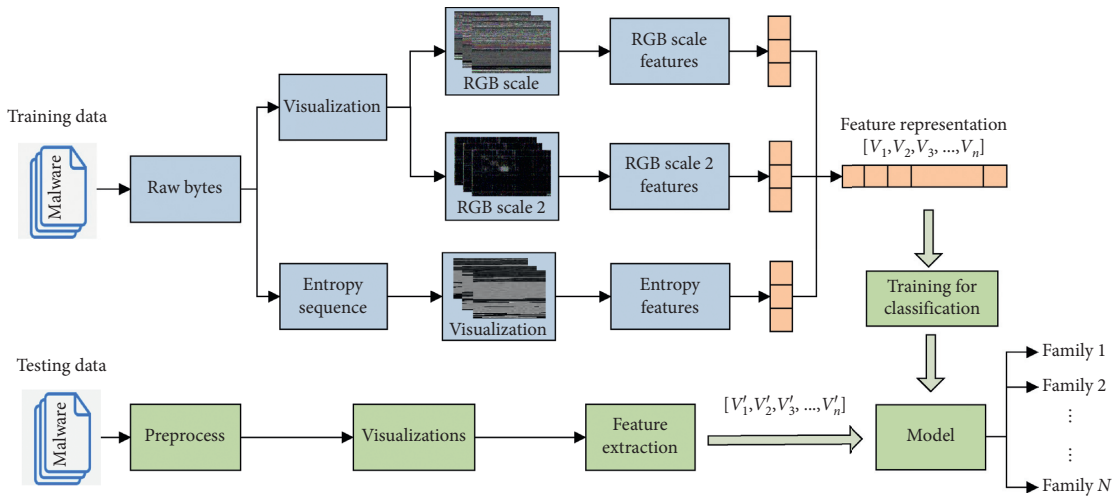


Figure 1: Overview of the method.

Specifically, for the byte sequence level RGB feature, we transform the raw bytes of malware into RGB image representation in bytes sequence order and extract the bottleneck features based on the convolutional layer of the deep learning network. For location information level RGB feature, we convert the malware into an RGB image representation using a visualization method containing location information. For entropy sequence feature, we first extract the entropy sequence of the malware, then convert the entropy sequence into a grayscale image representation, and extract the bottleneck features. Finally, a classifier is trained based on three types of features to classify malware.

3.2. The Byte Sequence Level RGB Feature. The features extracted from malware have a significant impact on classification performance. The method based on the malware visualization and texture features has been proved to be useful in classifying malicious code. In the field of image recognition, the deep learning model based on RGB images has better results than most traditional image recognition technologies. Therefore, this paper first extracts malware features based on the deep learning model and RGB images representation. This method mainly involves two steps: first, it transforms the malicious code into an RGB visual representation; second, it extracts the features contained therein. However, different malware visualization features would have different effects on the classification of malicious codes. In this section, we first visualize the malicious code based on byte sequence order and extract features as one of the critical elements of malicious code classification.

3.2.1. The Byte Sequence RGB Visualization Method. This paper visualizes the malware based on the raw bytes of malware samples. The raw bytes of the malicious code are a 1, 0-bit data stream, and each byte contains 8-bit data. Without processing malware samples, this paper directly extracts features based on the raw bytes of the malicious code to ensure that the malware image can contain all the features of the malicious code. At the same time, it can improve the efficiency of malware feature extraction and facilitate large-scale malicious code detection.

When the byte data sequence of the malware is obtained, it needs to be converted into RGB image representation, which involves two key issues: first, how to convert malicious code data into the pixel data of image; second, how to set the image format (i.e., height and width). For the first issue, the byte data of the malware are treated as 8-bit unsigned integer data, and the value range of each byte data is [0, 255]. The RGB channel data value range of each pixel is also [0, 255], and each pixel contains three channels of red, green, and blue. Therefore, every three bytes of data are combined into one group. According to the one-to-one correspondence relationship between each group of data and the three-channel data of pixels, the byte data sequence is converted into RGB image pixels in sequence order. The process is shown in Figure 2.

For the second issue, the range of malicious code file size is large. To ensure that the image has an appropriate aspect ratio, this paper formulates different image size setting standards based on the malware file size. The specific content is shown in Table 1.

The size of the malware is different, and the malware visualization image size is also different. To improve the performance of malware classification and facilitate the application of the deep learning algorithms, this paper normalizes the image to a uniform size after the transformation. As shown in Figure 3, there are some malware visualizations of malware samples. The image size of all malicious code samples is unified to $224 * 224 * 3$. It can be seen from the figure that the transformed images of malicious code samples of the same family have strong similarities, and there are some differences between malicious code samples of different families.

3.2.2. Feature Extraction of the Byte Sequence RGB Image. In the field of image processing, deep learning algorithms have been proven to handle various problems better than traditional feature extraction algorithms (such as GIST and LBP).

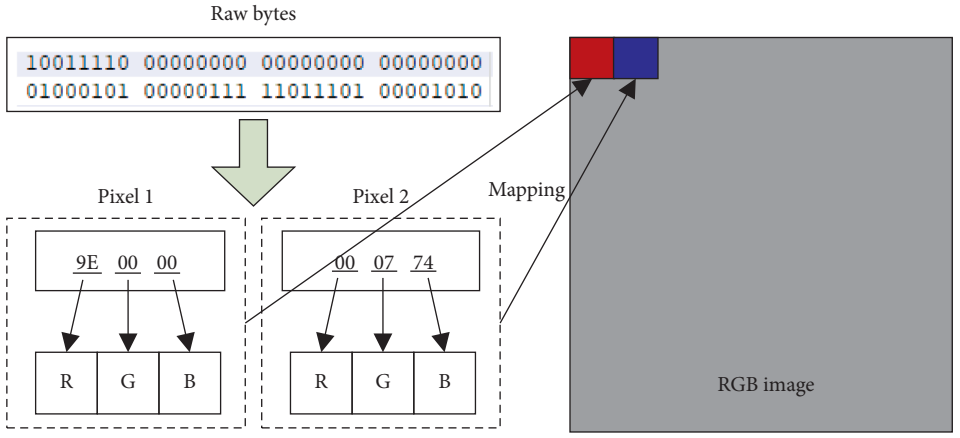
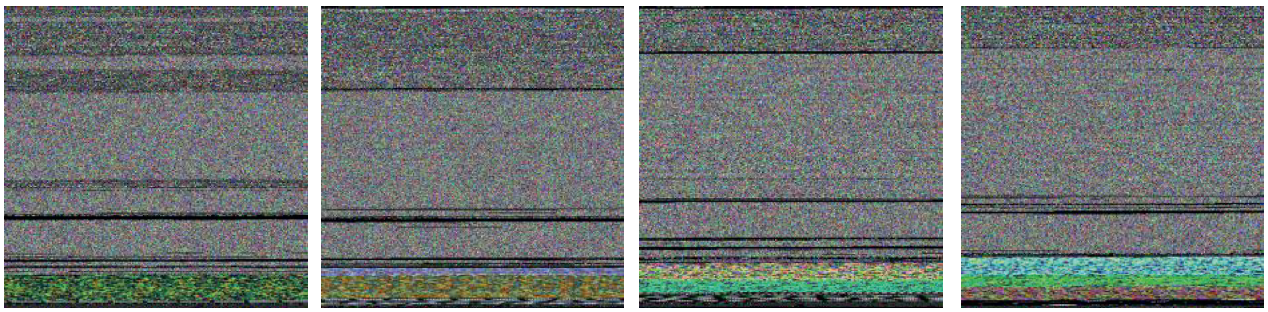


Figure 2: The transformation process of byte sequence RGB image.

Table 1: The byte sequence image conversion width.

File size (kB)	Image width	File size (kB)	Image width
<10	16	100–200	192
10–30	32	200–500	256
30–60	64	500–1000	384
60–100	128	>1000	512



(1) C2LOP.gen!g (2) C2LOP.P

Figure 3: The examples of byte sequence RGB visualization of malware samples.

Moreover, in malware classification, deep learning algorithms can obtain better classification results than texture feature extraction algorithms. Therefore, this paper does not use traditional texture feature extraction algorithms but processes malicious code images based on deep learning algorithms. However, the number of variants of different malware families is various. In some cases, the number of malicious code variants of some families is small (for example, newly emerged malicious code variants), which is not enough to form a training sample set, making the classifier unable to effectively learn the characteristics of the new family. Cui et al. [4] have pointed out that the classification performance of the deep learning classifier is not good enough when the malware samples of some families are insufficient. Therefore, this paper does not directly use deep learning algorithms but trains the classifier based on transfer learning algorithms. Combined with the characteristics of malware variant dataset, this paper trains the malware classifier based on the deep learning network that has been trained on a large-scale image dataset.

First, this paper extracts malicious code image features based on a deep network model. In recent years, some deep learning networks with better recognition effects have been proposed (e.g., ResNet50 [29], Xception [30], and Inception [31–34]). Different deep learning networks have different structures and different performances. The VGG16 deep network and VGG19 deep network [35] include 16-layer and 19-layer networks, respectively, which have achieved good performance in the field of image recognition. Rezende et al. [28] extracted the visualization features of the malware based on the VGG16 network and achieved good performance. However, as the network depth increases, the deep learning network would face some problems, such as the degradation problem [29], which refers to the fact that adding more layers to the suitably deep network would lead to higher

training errors. But these problems are address by the ResNet network. This makes it possible to train a network with deeper layers on the dataset, and the number of network layers could increase to more than 50, which has achieved better results than the VGG16 deep network in the field of image recognition. Besides, these deep networks have many different advantages and have achieved better performance than the VGG16 network in the field of image recognition. Therefore, these networks may be able to achieve better classification performance in malware classification. We evaluate the classification performance of these deep networks and finally extract byte sequence RGB image features based on the ResNet50 deep learning network.

The deep learning algorithms have achieved good performance in the field of image recognition, but as the network deepens, the application of deep learning networks will encounter two limitations: vanishing/exploding gradients problem [36–38] and degradation problem [29]. The problem of vanishing/exploding gradients has been addressed by normalized initialization [39, 40] and intermediate normalization layers [32]. However, the deep learning network still faces another problem: the degradation problem. As the network depth increases, the accuracy will gradually increase to saturation, and then, there will be a problem of rapid decline. However, the problem is not caused by overfitting [29]. In the same training round, a network with a deeper network (degraded network) has a higher error rate than a network with fewer layers [41]. But the degradation problem could be addressed by the application of the residual network. In the residual network, the residual unit is introduced, and its structure is shown in Figure 4.

In the residual network, the residual mapping is defined as the following:

$$H(x) = F(x) + x, \tag{1}$$

where $F(x)$ is the residual function. In the deep learning network, the additional residual unit connection method is called shortcut connection. In the residual network, this connection is defined as the following:

$$y = F(x, W_i) + x, \tag{2}$$

where $F(x, W_i)$ represents the residual mapping to be learned. For the example in Figure 4, the $F(x, W_i)$ in which σ represents ReLU [42], $F + x$ represents the shortcut connection. If the dimensions of X and $F(x)$ are different in the deep learning network (e.g., when changing the input/output channels), this connection is defined as the following:

$$y = F(x, W_i) + W_s x, \tag{3}$$

where W_s is a linear projection of the x to make it consistent with the dimension of $F(x)$. By importing residual values and residual units into the network, the correlation between the shallow and deep networks is enhanced, and the influence of the degradation problem is reduced. To evaluate the performance of the Reset50 network, we conducted a lot of experiments. The experimental results show that the residual deep learning network ResNet50 can effectively extract byte sequence RGB image features. The structure of the ResNet50 network is shown in Figure 5. The ResNet50 network includes 50 layers and about 3.8×10^9 parameters in total. Inspired by the transfer learning algorithms, we remove the fully connected layers of the network and use the convolutional layer in the ResNet50 network to extract the byte sequence RGB image features.

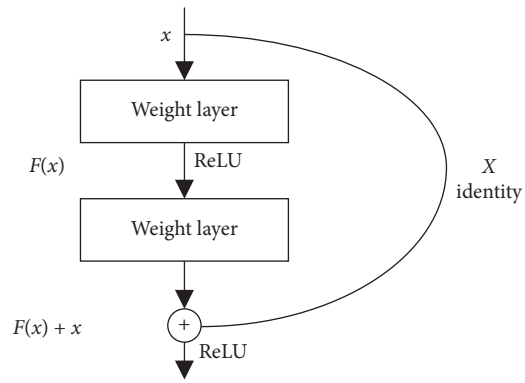


Figure 4: The residual unit.

3.3. The Location Information Level RGB Feature. The byte sequence level RGB image visualization method uses the raw byte data of the malicious code as the original data of the image pixels, and the locations of the pixels are arranged in byte order, without further considering the location information of the pixels. In this section, an image transformation method containing location information is proposed. The raw bytes of the malware are divided into two parts: location information and pixel information. In this way, the converted RGB image could contain richer information to enhance the performance of malicious code classification.

3.3.1. The Location Information Level RGB Visualization Method. In RGB images, the location of the pixel also has an essential influence on the characteristics of the image. In this section, the malware visualization is also based on the raw bytes of malware, which also involves two key issues: first, how to convert malicious code data into the pixel data of image; second, how to set the image format. In this section, the raw byte of the malware is still regarded as 8-bit unsigned integer data, and every five bytes of data are a group. The first two bytes of data in each group of data represent location information, and the last three bytes of data represents the pixel information of the RGB image. The process of location information level RGB image is shown in Figure 6.

Specifically, the value range of each byte is [0, 255], and we set the size of each image to $256 * 256$. The data coordinates of the bottom left corner of the image are set to (0, 0). All pixels in the image are initialized to 0. For each group of data, the first byte represents the X coordinate of the pixel in the image, the second byte represents the Y coordinate, and the last three bytes of data respectively represent the values of the R, G, and B channels of each pixel in order. Then, each group of data can be converted into a pixel in the image. The

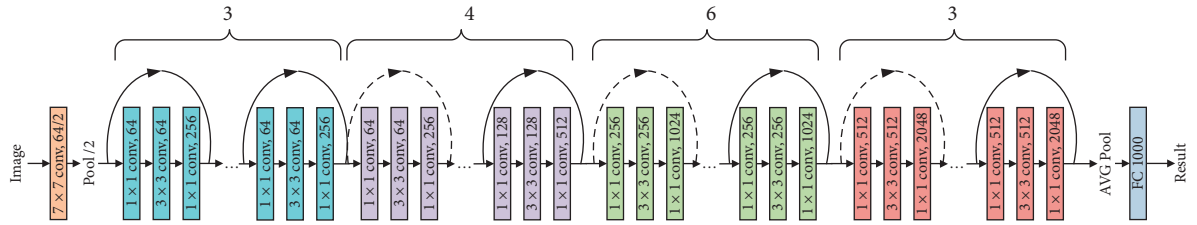


Figure 5: The structure of the ResNet50 network.

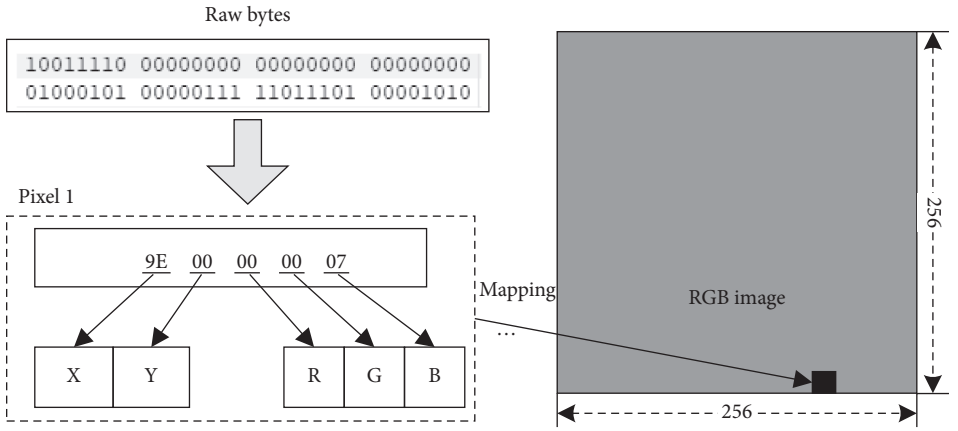


Figure 6: The transformation process of location information RGB image.

The RGB image M_i contains the information of the malware feature, and $(X_m, Y_m, R_m, G_m, B_m)$ is a group of raw byte data of malware to be processed.

Input: image M_i, $(X_m, Y_m, R_m, G_m, B_m)$

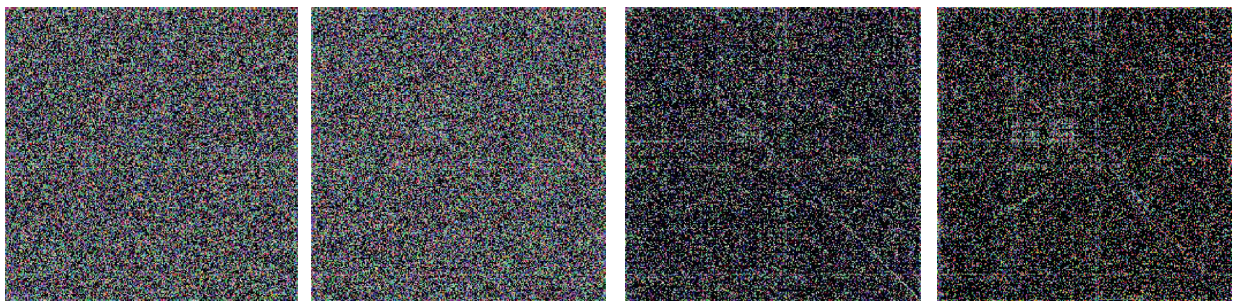
(1) Find the pixel (R_i, G_i, B_i) to the coordinates (X_m, Y_m) in M_i
(2) $R_i = (R_i + R_m)\%255$
(3) $G_i = (G_i + G_m)\%255$
(4) $B_i = (B_i + B_m)\%255$
(5) Modify the pixel data of (X_m, Y_m) in M_i to (R_i, G_i, B_i)
(6) **Return:** image M_i

Algorithm 1: Image pixel transformation.

image pixel processing method is shown in Algorithm 1. Each group of data is added to the pixel data of the corresponding coordinate separately, and the sum value is used as the final data of the image pixel. If the data are greater than 255, the 255 remainder operation is performed on the data.

The size of the RGB image is $256 * 256$, and there is no need to perform standardized operations on these images. Some examples of the location information level RGB image are shown in Figure 7. It can be seen that the location information RGB level image seems to contain "noise," and there are not many obvious blocks in the image. However, some images still include noticeable lines and blocks. As shown in Figure 7, the samples of similar families C2LOP.gen!g and C2LOP.P exhibit different image features, so it can be inferred that some similar family samples could be classified based on these image characteristics.

3.3.2. Feature Extraction of the Location Information RGB Image. The location information level RGB image shows a big difference from the byte sequence level image, and the performance of different deep learning networks is also different. The ResNet network makes it possible to train a network with deeper layers on the dataset, and the number of network layers even could increase to 200, which has achieved good performance. It has also been found that different convolution kernels have different performance on image recognition. In the process of image recognition, it is



(1) C2LOP.gen!g (2) C2LOP.P

Figure 7: The examples of location information RGB visualization of malware samples.

of great significance to quickly find the best convolution kernel, and the problem is addressed by the Inception network and Xception network. They use multiple filters of different sizes to make the network "wider." For the same image, different convolution kernels extract different features, so through the network training process, the computer can automatically adjust the weight for the image to find the optimal feature extraction method. This method enables the deep learning network to achieve better performance under the same depth. Therefore, the Xception deep network may achieve better performance than VGG16 deep network in malicious code classification. Based on this hypothesis, we evaluate the classification performance of these deep networks and finally extract the location information level RGB image features based on the Xception deep learning networks.

Xception network is another improved model after the Inception V1, V2, and V3 models proposed by Google. Compared with the traditional convolutional neural network, the Inception network model imports the Inception module. It uses multiple filters of different sizes (including $1 * 1$ convolution, $3 * 3$ convolution, $5 * 5$ convolution, and maximum pooling) to make the network "wider.. For the same image, different convolution kernels extract various features; so, through the network training process, the computer can automatically adjust the weight for the image to find the optimal feature extraction method. This method enables the deep learning network to achieve better performance under the same depth. Besides, the Xception network uses the depth-wise separable convolution module in the Inception network to further enhance the performance of the deep learning network. As shown in Figure 8, it is an extreme version of the Inception model. For the input, a $1 * 1$ convolution kernel is used for convolution operation to obtain N channel data, and then, N $3 * 3$ convolution kernels are used to perform the convolution operation. Each channel data are convolved so that each $3 * 3$ convolution kernel is convolved with one channel. At last, all the results are merged as the output.

The depth-wise separable convolution used in Xception is different from the Inception network in the following two points: (1) the order of operations is different. The Xception network first uses M $3 * 3$ convolution kernels to convolve the input data. It then uses N $1 * 1$ convolution kernels and M output results to perform convolution operations to

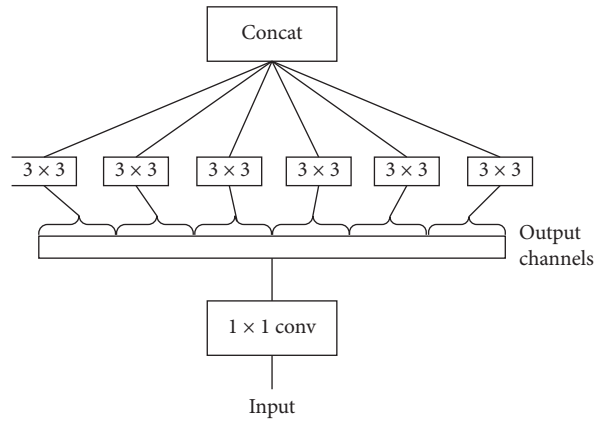


Figure 8: An extreme version of Inception module.

generate N results. However, the Inception network first performs a $1 * 1$ convolution operation. (2) All processes in the Inception network are followed by nonlinear operations (i.e., ReLU). However, most of the depth-wise separable convolution does not require nonlinear operations. The improvements of these two aspects have enabled the Xception network to have a better performance in the field of image recognition. Moreover, the residual module is also imported into the Xception network. The Xception network has 36 convolutional layers for extracting image features, which also uses multiple residual module connections. At the end of the Xception network, a fully connected layer and a logistic regression layer are used to classify images. The network structure is shown in Figure 9. In the figure, "Spconv" is the abbreviation of SeparableConv, which refers to the use of a depth-wise separable convolution module. We remove the fully connected layer in the Xception network and extract the features of the image based on the convolutional layers of the Xception. These features are the second part of the input features of the malicious code variant classifier.

3.4. The Entropy Sequence Feature. The entropy distribution of the raw byte data of malicious code is also one of the essential characteristics of malicious code. In related research, various entropy feature processing methods (e.g., linear features, the bag of words model) have achieved good

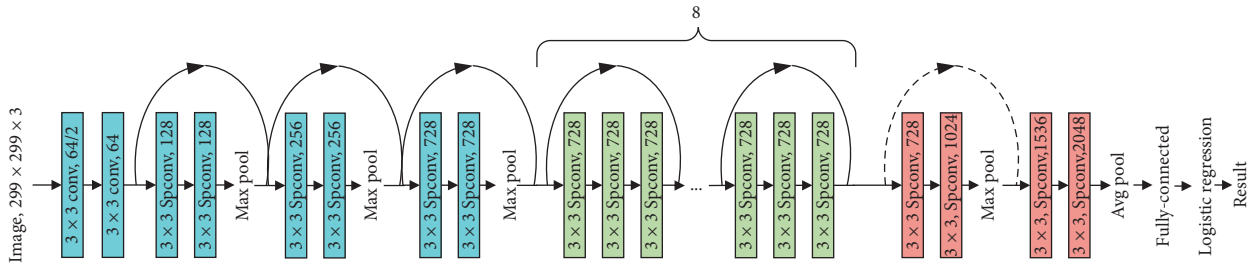


Figure 9: The structure of the Xception network.

results in the field of malicious code detection. Similar to the development of image recognition technology, deep learning algorithms enable the network to automatically learn the features contained in images and can often achieve better results. Therefore, we first extract the entropy sequence of the malware, then visualize the entropy, and process the entropy visualization based on the deep learning algorithms to extract features for malware classification.

3.4.1. Entropy Sequence Extraction. The entropy is a measure of the randomness and uncertainty of data distribution. To compute the entropy sequence of the malware, we first divide the raw bytes of the malware into continuous data blocks (the data are represented in hexadecimal: 00h-FFh), then compute the entropy of each block, and finally connect the entropy of each block according to the order of the blocks to form the entropy sequence. The value range of each byte is [0, 255]. It is crucial to ensure that all data in the block can be used to compute the entropy value. So, the size of the block is set to 256. When computing the entropy sequence, if the length of the last block is less than 128 bytes, the block will be discarded. Otherwise, the block is supplemented with data zero to make its length reach 256.

For each block, the method of computing entropy is as follows:

$$H(X) = -\sum_{i=0}^{255} p(x_i) * \log 2\, p(x_i), \tag{4}$$

where x_i represents a specific raw byte value and p_i represents the probability (frequency) of this value in the block, $H(x)$ represents the entropy value of the block, and the range of its value is zero to eight. When all bytes in the block are equal, the value of entropy is zero. If all the values in the block are different, the value of entropy is eight. If the raw byte of the malware is divided into N blocks, we represent the entropy sequence as $H_s = h_1, h_2, h_n$. Some examples of the entropy sequence are shown in Figure 10. Figures 10(a) and 10(b) show the entropy sequence of the malware samples of family Rbot!gen. Figures 10(c) and 10(d) show the entropy sequence of the malware samples of family Adialer.C. It can be seen that the entropy sequences of the same malware family samples are very similar, but the entropy sequence distributions of different family samples are quite different.

3.4.2. Entropy Sequence Visualization. This paper visualizes the entropy sequence, which also involves the two critical issues mentioned above: first, how to convert malicious code data into the pixel data of image; second, how to set the image format. For the first issue, the value range of the entropy data is [0, 8], and the value range of image pixels is [0, 255]. Therefore, this paper amplifies the entropy value and processes each entropy value according to the following formula:

$$P_{h_i} = 2^{h_i} - 1, \tag{5}$$

where h_i represents the entropy value of the block i in the malware, P_{h_i} represents the enlarged value, and its value range is [0, 255], which is the same as the value range of image pixels. This paper uses P_{h_i} as the pixel value of the entropy image. For the second issue, this paper uses the grayscale visualization method to process the entropy sequence. The image format of the entropy sequence is shown in Table 2.

Moreover, the length of the entropy sequence of the malware is different, and the size of the visualization image is also different. To enhance the classification performance, this paper also standardizes the grayscale entropy image and modifies the image format to the same size. Some grayscale images of the entropy sequence are shown in Figure 11. The image size is unified to $224 * 224$. It can be seen that the entropy images of malware samples of the same family have strong similarities, but the images of different families have different features.

3.4.3. Entropy Grayscale Image Feature Extraction. Several scholars have achieved good performance in extracting grayscale image features based on deep learning algorithms. Therefore, this paper also extracts the features of the entropy grayscale images based on the deep learning algorithms. After several experiments, the ResNet50 deep learning network introduced in Section 3.2 could effectively extract the features of the entropy image. In this section, the features of the grayscale image of the entropy sequence are also extracted based on the ResNet50 deep learning network which is used as one of the input vectors of the final malicious code classifier.

We extract features from the byte sequence RGB image and the location information RGB image. Besides, we also extract the entropy features from the raw bytes of the

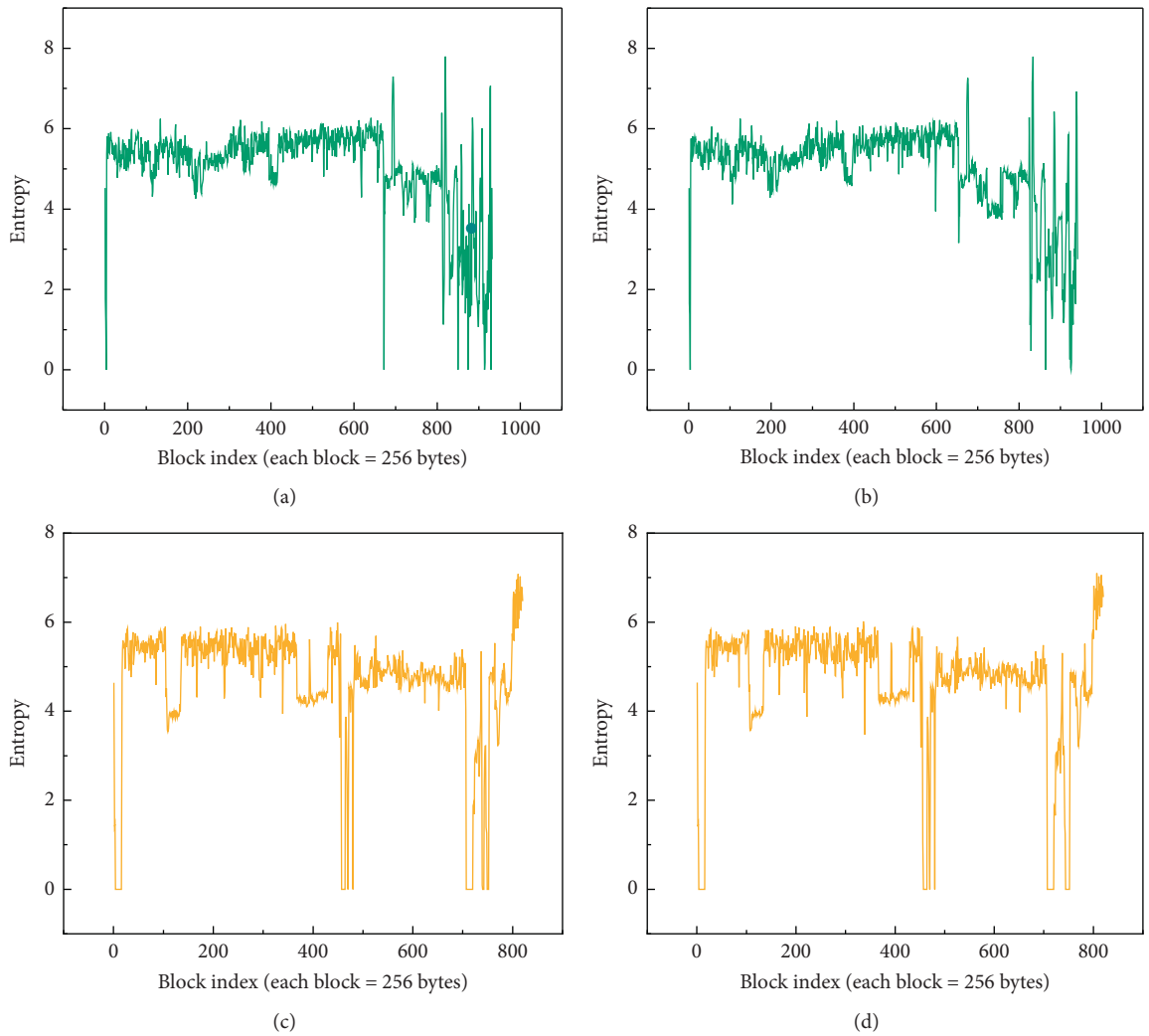


Figure 10: The entropy sequences of malware samples from different families. (a) Sample 1 of family Rbot!gen. (b) Sample 2 of family Rbot!gen. (c) Sample 3 of family Adialer.C. (d) Sample 4 of family Adialer.C.

Table 2: Entropy sequence image conversion width.

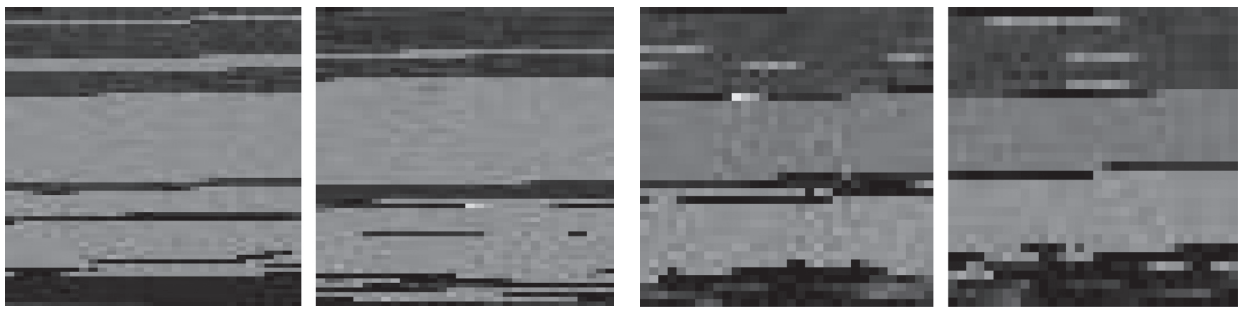
Sequence (KB)	Image width	Sequence (KB)	Image width
<10	32	100–200	384
10–30	64	200–500	512
30–60	128	0.500–1000	768
60–100	256	>1000	1024

malicious code. Finally, these three types of features are combined to train a machine learning model for malware classification.

4. Evaluation

4.1. Implementation and Setup. To evaluate the performance of the proposed method, we implemented a prototype system. The system is programmed in Python 2.7. The deep learning part of the system is programmed by the Keras library, and the machine learning part is programmed by the Sklearn library. The system is deployed on a PC with an Intel(R) Xeon(R) Gold 6139 CPU (2.3 GHz, 72 cores), a TITAN RTX graphics card and 188 GB RAM.

4.1.1. Dataset. To evaluate the proposed method, our experiments are conducted based on two malware datasets: malimg dataset [3] and Big 2015 dataset [43]. The malimg dataset consists of 9,339 malicious samples from 25 families. All the malicious samples are classified by the Microsoft security platform. The detailed distribution of the samples in



(a) (b)

Figure 11: The examples of byte sequence RGB visualization of malware samples. (a) C2LOP.gen!g. (b) C2LOP.P.

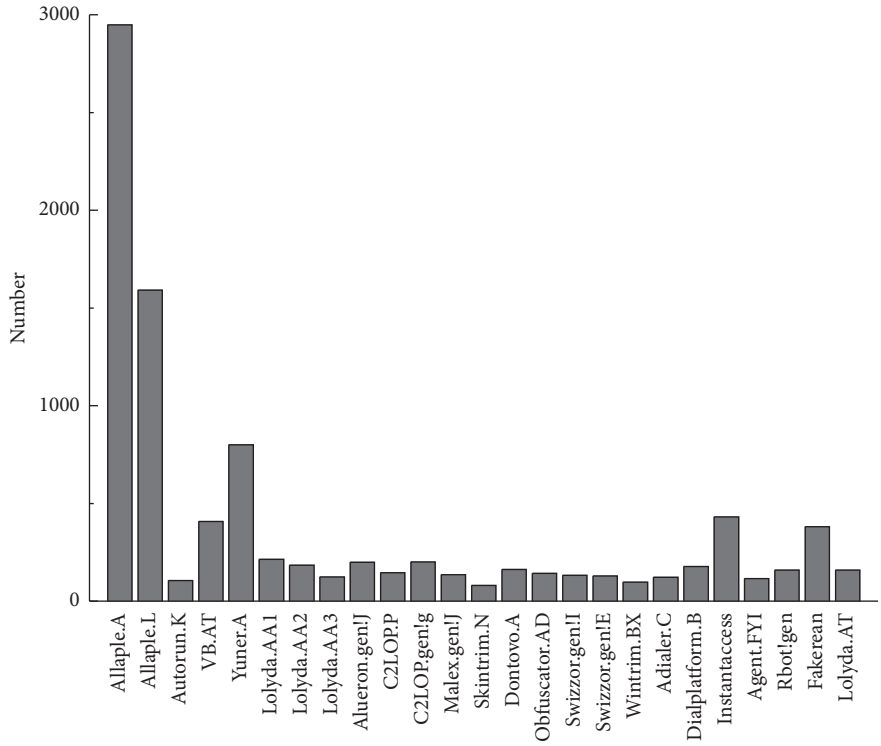


Figure 12: The distribution of samples in the dataset.

the dataset is shown in Figure 12. The big 2015 dataset is provided by the Microsoft Malware Classification Challenge (BIG 2015) [43]. The dataset consists of 10,868 samples from 9 families (Ramnit, Lolipop, Kelihos_ver3, Vundo, Simda, Tracur, Kelihos_ver1, Obfuscator.ACY, and Gatak). All samples are provided by Microsoft Research and Northeastern University.

To ensure the accuracy and reliability of the experimental results, this paper uses a 10-fold cross-validation method. In the experiment, the dataset is divided into ten parts. In each test process, nine parts are selected as the training set, and the remaining one part is used as the test set. A total of ten experiments are conducted, and finally, the results of all experiments are combined to evaluate the method.

4.2. Evaluation Metric. The evaluation matric in this paper includes accuracy, precision, recall, receiver operating characteristic (ROC) curve, and area under ROC (AUC) curve. Accuracy: the proportion of the number of correct classification of malware samples to the entire samples. Precision: the ratio of true positive samples to the positive samples classified by the classifier. Recall: the proportion of true positive samples to entire positive samples in the dataset. F1-measure is the weighted harmonic average of precision and recall. ROC curve: the ordinate of the ROC curve is the true positive rate and the abscissa is the false positive rate. The true positive rate is also called recall. When evaluating the generalization ability of two classifiers, the area under the ROC curve (AUC) is usually compared. The size of the AUC area represents the generalization ability of

the classifier. The related definition formulas are shown as follows:

$$\begin{aligned}
\text{Acc} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}},\\
\text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}},\\
\text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}},\\
\frac{1}{F_1} &= \frac{1}{2} \times \left(\frac{1}{P} + \frac{1}{R}\right),\\
F_1 &= \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}},\\
\text{FPR} &= \frac{\text{FP}}{\text{TN} + \text{FP}},
\end{aligned} \tag{6}$$

where the true positive (TP) samples refer to the malware samples in which malicious code samples are correctly classified into their corresponding families. The false positive (FP) samples refer to the malware samples in which the samples are misclassified into the specified family. The true negative (TN) samples refer to the malware samples which do not belong to the specified family and are indeed classified into other families. The false negative (FN) samples refer to the malware samples of the specified family which are misclassified into other families.

4.3. The Classification Performance of Three Kinds of Visualization Features. We first evaluated the classification performance based on different deep learning networks. Then, we extracted the visualization features and further evaluated the performance of three types of features based on different machine learning algorithms. The experiments in this section are conducted based on the malimg dataset.

4.3.1. The Classification Performance of the Byte Sequence RGB Image Features. To evaluate the classification performance of different deep learning networks, we first extract malware byte-sequence visualization features based on nine deep learning networks (i.e., VGG16, VGG19, ResNet50, DenseNet, EfficientNet, InceptionResNet, InceptionV3, Xception, and NasNet). Then, we train the classifier based on the random forest algorithm. In the setting of each network, all the deep learning networks use the default configuration parameters in Keras. The classification performance is shown in Figure 13.

It can be seen that, except for the poor performance of the EfficientNet deep learning network, the other deep learning networks can extract the features of malicious code and achieve good classification performance. In the experiment, all networks remove the last fully connected layer, and the VGG16 and VGG19 networks also remove the pooling layer after the convolutional layer (if only the last fully connected layer is removed, the performance of malware classification will be poor). However, this leads to a high dimension of image features extracted by the VGG16 and VGG19 networks (25088 and 100352, respectively). The experimental results of ResNet50 and DenseNet201 networks are almost the same. In this paper, the ResNet50 network is temporarily selected for extracting byte sequence level RGB image features.

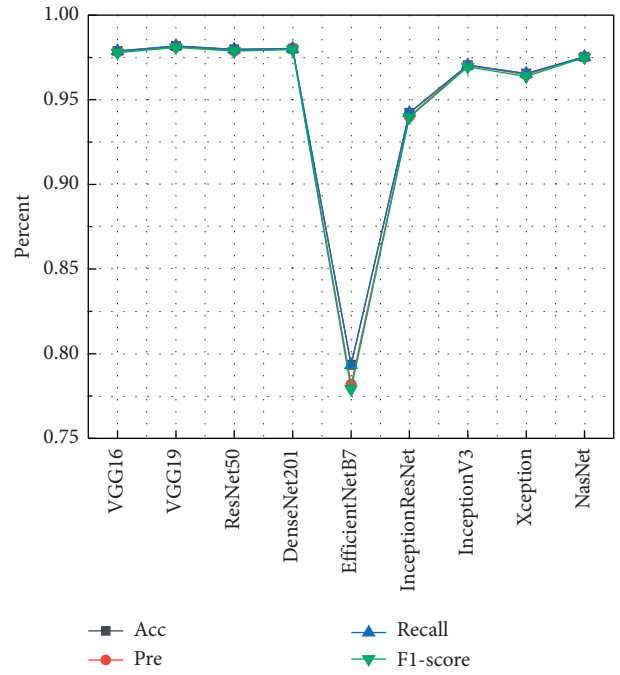


Figure 13: The performance of different deep learning networks trained based on byte-sequence features.

We extract byte sequence level RGB image features based on the ResNet50 network. Then, we conduct experiments to evaluate the performance of different machine learning classifiers, which are trained based on random forest (RF), multilayer perceptron (MLP), k-nearest-neighbor (KNN), support vector machines (SVM), decision tree (DT), and Gaussian Naive Bayesian (NB) algorithms. Each sample has 2,048 features that are extracted by the ResNet50 network. In the setting of each classifier, the K value of the KNN classifier is set to 2, the SVM uses a linear kernel function, and the remaining classifiers use the default configuration parameters in SKlearn. The results are shown in Table 3.

From Table 3, we can see that the performance of the MLP classifier is relatively poor, which may be due to the default configuration. The classification performance could be improved by adjusting the parameters. KNN, RF, and SVM classifiers achieved relatively better classification performance, all achieving about 98% classification accuracy. In the experiment, the SVM classifier has the best classification performance, and the classification accuracy rate can reach 98.87%, but its training time is relatively long (i.e., about 60 s). However, the running time on the test set is about 4 seconds, and the efficiency is still relatively high, which proves that it is still suitable for large-scale malicious code classification.

Table 3: The performance of byte-sequence-feature-based classifiers.

Classifier	Training time (s)	Acc (%)	F1-score (%)	AUC
RF	2.64	97.97	97.87	0.9965
KNN	1.53	98.35	98.35	0.9890
MLP	157.90	90.36	88.91	0.9928
SVM	**61.90**	**98.87**	**98.84**	**0.9996**
DT	14.41	96.47	98.84	0.9559
NB	0.226	96.83	96.97	0.9813

The significance of the bold values given in the table is that, in the experiment, the SVM classifier achieved the best classification performance, and the classification accuracy rate can reach 98.87%.

4.3.2. The Classification Performance of the Location Information RGB Image Features. To evaluate the classification performance of different deep learning networks, we also extract malware location information visualization features based on nine different deep learning networks and then train the classifier based on the random forest algorithm. In the setting of each network, all the deep learning networks use the default configuration parameters in Keras. The classification performance is shown in Figure 14.

It can be seen that the classification performance of the classifiers trained based on the location information level RGB image features is worse than classifiers trained based on the byte sequence level RGB image features. However, the classification performance of the classifier trained based on the features extracted by the DenseNet201, InceptionV3, Xception, and NasNet networks could still reach 90%.

Inspired by the experiment in the previous section, we also evaluate the classification performance of the classifier trained based on these four types of features and the support vector machine (SVM) algorithm. Among them, the NasNet network extracts the features based on the linear kernel function, and the optimal solution cannot be trained. The performance of the other three classifiers is as follows: the accuracy of the DenseNet201 network features could reach 95.58%, the accuracy of the InceptionV3 feature could reach 94.67%, and the accuracy of the Xception features could reach 95.65%. It can be seen that the feature classification extracted by the Xception deep learning network has the best performance. Therefore, in this paper, the Xception network is temporarily selected for extracting location information level RGB image features.

We extract location information level RGB image features based on the Xception network. Then, we also conduct experiments to evaluate the performance of different machine learning classifiers. Each sample has 2,048 features that are extracted by the Xception network. In the setting of each classifier, the K value of the KNN classifier is set to 2, the SVM uses a linear kernel function, and the remaining classifiers use the default configuration parameters in SKlearn. The results are shown in Table 4.

From Table 4, we can see that the performance of the classifiers trained based on the location information level features is poor than the byte-sequence-feature-based classifiers. Among them, the NB classifier has the worst performance, and its accuracy is less than 70%. The classification performance of KNN, RF, and SVM classifiers is still relatively better than the others, all of which have achieved a classification accuracy of more than 90%. Among them, the SVM classifier has the best classification performance, and the classification accuracy can reach 95.65%. But in the experiment, the training time of the SVM classifier is still a bit long, but the running time on the test set is still about 4 seconds, and it still has practical application value.

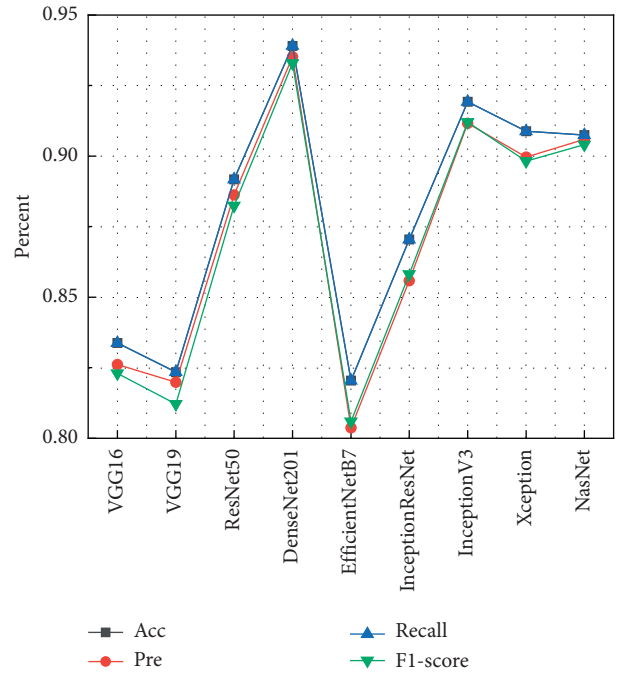


Figure 14: The performance of different deep learning networks trained based on location-information features.

Table 4: The performance of location-information-feature-based classifiers.

Classifier	Training time (s)	Acc (%)	F1-score (%)	AUC
RF	1.20	90.88	89.82	0.9843
KNN	1.61	91.91	91.78	0.9632
MLP	134.78	81.25	77.55	0.9835
SVM	**67.94**	**95.65**	**95.66**	**0.9989**
DT	10.19	86.28	86.35	0.9021
NB	0.10	69.70	68.12	0.9641

4.3.3. The Classification Performance of the Entropy Visualization Features. From the previous experiment, it can be seen that the classification performance of classifiers trained based on the features extracted by the ResNet50 network is all relatively good. Therefore, this paper directly extracts the entropy visualization features based on the ResNet50 network and conducts experiments to evaluate the performance of classifiers trained based on different machine learning algorithms. Each sample has 2,048 features that are extracted by the ResNet50 network. The setting of each classifier is the same as the previous experiment. The K value of the KNN classifier is set to 2, the SVM uses a linear kernel function, and the remaining classifiers use the default configuration parameters in SKlearn. The results are shown in Table 5.

Table 5: The performance of location-information-feature-based classifiers.

Classifier	Training time (s)	Acc (%)	F1-score (%)	AUC
RF	2.07	86.56	86.77	0.9935
KNN	1.43	86.00	85.94	0.9828
MLP	91.62	82.79	81.99	0.9917
SVM	**144.27**	**86.71**	**86.82**	**0.9957**
DT	15.03	84.42	84.35	0.9514
NB	0.21	85.57	85.82	0.9707

Table 6: The performance of combined-feature-based classifiers.

Classifier	Training time (s)	Acc (%)	F1-score (%)	AUC
RF	2.36	99.15	99.14	0.9997
KNN	7.25	99.10	99.09	0.9917
MLP	118.56	88.03	85.39	0.9902
SVM	**120.02**	**99.73**	**99.73**	**0.9999**
DT	26.55	98.13	98.11	0.9765
NB	0.64	98.68	98.68	0.9895

From Table 5, we can see that the performance of the classifiers trained based on the entropy visualization feature is poor than the previous two feature-based classifiers. The classification accuracy of each classifier is about 85%, among which the KNN, RF, and SVM classifiers still have a relatively better classification performance, and the classification accuracy could reach 86%. The classification performance of the SVM classifier is still the best, and the classification accuracy can reach 86.71%. Analyzing the experimental results, the reason for the decrease in classification accuracy is mainly due to the poor classification effect for similar families such as Allaple.L and Allaple.A. However, it has achieved a better classification performance for the C2LOP.P and C2LOP.gen!g and Swizzor.gen!j and Swizzor.gen!I families, which is somewhat different from the classification effect of the two types of features as mentioned above.

4.4. The Classification Performance of the Combined Features. Based on visualization technology, this paper extracts three types of features of malware: byte-sequence features, location-information features, and entropy visualization features. The length of the three types of features is 2,048. To evaluate the impact of the three types of features on the classification and whether the combination of the three types of features can help improve the experimental performance, this paper first trains different machine learning classifiers based on the combined features, with the same parameter settings as the previous experiment. The classification performance is shown in Table 6.

It can be seen from Table 6 that the classifier based on combined features performs very well. Except that the parameter settings of the MLP classifier may have problems, resulting in its poor classification performance, the other classifiers all achieved acceptable performance. Among them, the classification accuracy of the RF, KNN, and SVM classifiers have reached more than 99%, and the classification accuracy of the SVM classifier even could reach 99.73%. To further compare the classification performance of each type of feature and the combined feature, this paper conducts more experiments. The result is shown in Table 7.

In Table 7, RGB1 refers to byte-sequence features, RGB2 refers to location information level features, and Ent_gray refers to entropy grayscale image features. R1, R2, and Ent are the abbreviations for the three types of features, respectively. From the results, we can see that the classifiers trained based on three kinds of features all can achieve good malicious code classification performance, and the classification accuracy of the classifier trained based on byte sequence level features can reach 98.865%. Therefore, we first evaluated the performance of the classifiers based on the byte-sequence features combined with other two kinds of features separately and then evaluated the classification performance based on the combination of three types of features. From the experimental results, we can see that both the location information visualization feature and the entropy feature can improve the classification performance.

Moreover, the classification accuracy of the classifier trained based on the combination of three types of features has been further improved, and the classification accuracy can reach 99.732%. It can be seen that the three types of features all contribute to the final classification performance. To further evaluate the impact of different features on the classification performance, this paper compares the classification performance of different classifiers in each malware family, and the result is shown in Figure 15.

It can be seen from Figure 15 that among the three types of features, the classification performance of the byte-sequence-feature-based classifier is relatively better, but its classification performance on C2LOP.P and C2LOP.gen!g and Swizzor.gen!E and Swizzor.gen!I families is worse than the other two features. However, the classification performance of the combined features is best. Analyzing the misclassified samples, similar family samples still have similar characteristics, resulting in relatively low classification performance in the Swizzor.gen!I and C2LOP.P families. This issue needs further research in the future. Overall, this paper solves most of the sample classification confusion problems of similar families by combining features, and the application of combined features has better classification performance and practical value than single features.

4.5. The Classification Performance of Different Image Formats. The format of the image influences the texture characteristics of the image. In this section, we conduct experiments to evaluate the influence of image format. The malicious code classification method studied in this paper is developed on the traditional method of extracting image GIST and other texture features. It is difficult to process malicious code images of different formats based on the deep learning network. Therefore, we extract the GIST texture features of the malicious code images for classification to evaluate the impact of the changes in the image format on the malicious code classification performance. In the experiment, the format of malicious code images is set to 8 different formats (original, $32 * 32$, $64 * 64$, $128 * 128$,

Table 7: The performance of classifiers trained based on different visualization features.

Features	Acc (%)	Pre (%)	Recall (%)	F1-score (%)
RGB1	98.865	98.870	98.864	98.847
RGB2	95.653	95.755	95.655	95.662
Ent_gray	86.712	87.127	86.713	86.821
R1 + R2	99.539	99.560	99.539	99.537
R1 + Ent	99.615	99.634	99.614	99.609
R1 + R2 + Ent	**99.732**	**99.743**	**99.733**	**99.730**

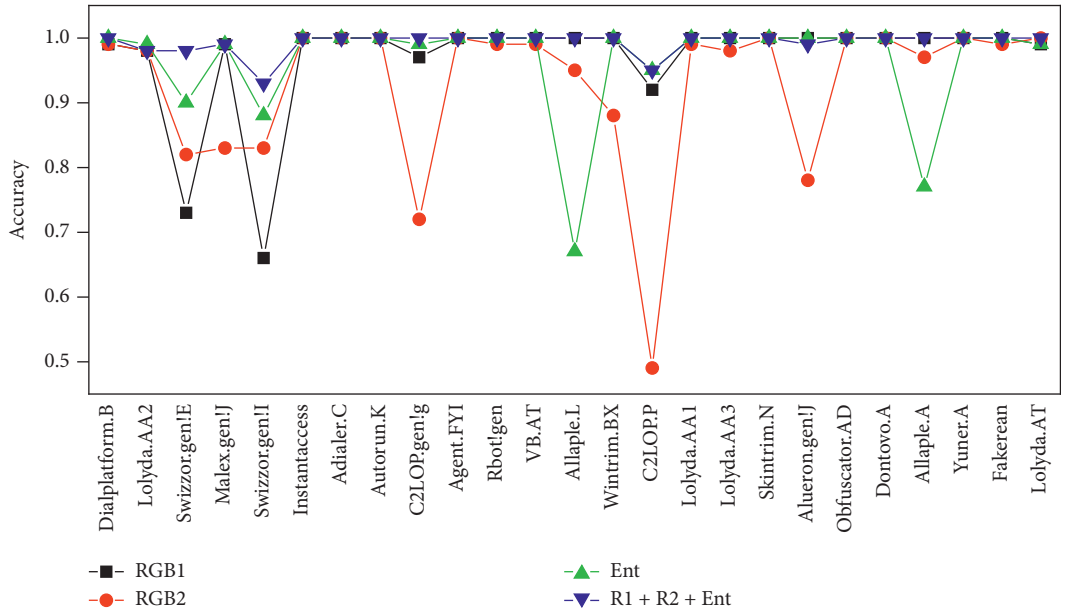


Figure 15: The performance of different visualization features on each family.

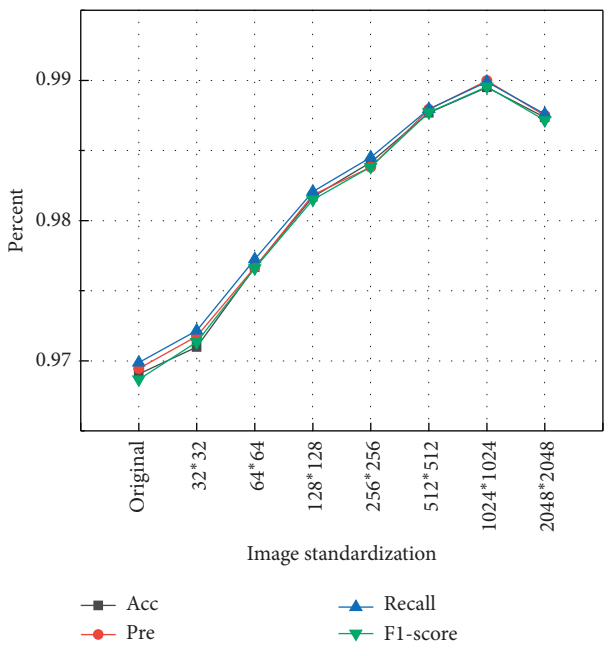


Figure 16: The performance of different image formats.

256 ∗ 256, 512 ∗ 512, 1, 024 ∗ 1, 024, and 2, 048 ∗ 2, 048). The experimental results are shown in Figure 16.

It can be seen from Figure 16 that the uniform and standardized formats of malicious code images can improve the accuracy of malicious code detection. The reason may be that the sizes of malicious codes in the same family are different, resulting in different sizes of converted images. In this way, there are some differences in the extracted image features, which may cause some confusion in the classification of malicious codes in different families. After the image format is unified, the difference in characteristics of malicious codes of the same family is reduced, and the classification performance is improved. At the same time, as shown in the figure, the classification performance is best when the malicious code image format is set to 1, 024 ∗ 1, 024. However, the image format is modified to a larger size, and the more time it takes to extract image features. When the image format is set to 256 ∗ 256, the feature extraction time of the entire dataset is about 1-2 minutes, and when the image is set to 1, 024 ∗ 1, 024, the experimental time for extracting GIST texture features is more than half an hour. Although the classification accuracy has been improved, the efficiency is too low and

needs to be improved in practical applications. Therefore, in our method, we resize all the malware images into a uniform size, which has improved the classification performance.

4.6. Compared with Other Malware Classification Methods. The classification technology based on the malware visualization features is a hot spot in recent research, and there have been many significant research results. To evaluate the proposed method, we compare the method with four famous methods proposed in related research in recent years (GIST + KNN [3], LBP + KNN [25], GIST + DSIFT + KNN [6], and VGG16 (fine-tune) [28]) on the malimg dataset. The experimental results are shown in Table 8. It can be seen that the method proposed in this paper has the best classification performance.

Naeem et al. [6] proposed a new method (GIST + DSIFT) to extract more complex features and improved classification performance. Besides, Rezende et al. [28] proposed a method (VGG16 (fine-tune)) to extract image features based on deep learning and transfer learning algorithms, achieving a better performance. These methods partially alleviate the problem of malicious code confusion in similar families. However, it can be seen that this paper has achieved the best classification accuracy based on three types of visualization methods and almost solved the problem of malware confusion of similar families in the malimg dataset. However, for the efficiency, due to the large number of features proposed in this paper, the running time of the proposed method is a bit longer. The training time of the experiment is about 120 seconds and the running time on the test set is about 7 seconds, but it is still applicable to the actual environment and can be used to detect large-scale malicious code samples. To sum up, the method proposed in this paper has achieved a good classification performance on the malimg dataset, showing its superiority to the other methods.

All the previous experiments were performed on the malimg dataset. To further evaluate the performance of the proposed method, we also conducted experiments to compare our method with the other four methods on the big 2015 dataset. The experimental results are shown in Table 9. It can be seen from Table 9 that the method proposed in this paper still shows the best classification performance. The training time on the Big2015 dataset is about 195.85 seconds, and the running time is about 7 seconds. It is still applicable to the actual environment.

In the experiment, VGG16 (fine-tune) has a better classification performance than the other three methods, which proves that the deep learning algorithm can extract better features in different datasets. The three types of features proposed in this paper describe the malware samples more detailed and achieve better classification performance. We have conducted further research on samples that were misclassified by this method. The proposed method only has a classification accuracy of 88% for Simda family samples, while the classification accuracy of other families is almost 100%. It could be found that the proposed method misclassifies some malware samples of the Simda family into Vundo and Obfuscator.ACY families. The misclassified samples have similar visual features with the samples of the two families, which leads to the misclassification. In future research, more in-depth research will be conducted on this issue.

Table 8: Proposed method compared with other malware classification methods on the malimg dataset.

Methods	Acc (%)	Recall (%)	F_1(%)	RT (s)
GIST + KNN [3]	97.28	97.28	96.61	0.423
LBP + KNN [25]	97.87	97.88	97.86	0.052
GIST + DSIFT + RF [6]	98.53	98.53	98.49	0.026
VGG (fine-tune) [28]	98.84	98.84	98.82	59.50
Our method	**99.73**	**99.73**	**99.73**	**7.625**

RT means running time.

Table 9: Proposed method compared with other malware classification methods on the Big2015 dataset.

Methods	Acc (%)	Recall (%)	F_1 (%)	RT (s)
GIST + KNN [3]	94.82	94.81	94.75	0.621
LBP + KNN [25]	91.87	91.88	91.87	0.045
GIST + DSIFT + RF [6]	95.09	95.10	95.06	0.004
VGG16 (fine-tune) [28]	97.92	97.93	97.92	62.59
Our method	**99.54**	**99.53**	**99.53**	**7.91**

RT means running time.

Moreover, to further evaluate the classification performance of the proposed method, we deeply compared the classification performance with the method proposed by Cui et al. [4] and Rezende et al. [28] on the malimg dataset for each family. The result is shown in Figure 17. The two methods proposed by Cui et al. [4] and Rezende et al. [28] used the raw bytes of malware and deep learning algorithms to achieve malicious code classification. Cui et al. [4] pointed out that training the classifier directly based on the deep learning algorithm and the malimg dataset is not a well solution, because the samples of some families in the malimg dataset is not enough for training, resulting in low classification accuracy. To improve the performance of the classifier, they fine-tuned the malware samples in the dataset. and Rezende et al. [28] trained the malware classifier based on the transfer learning algorithm and also achieved good classification performance.

The misclassification problem of similar families has always been a relatively difficult problem to solve in related research. In the malimg dataset, several methods have not been able to solve the misclassification problem of Swizzor.gen!E family and Swizzor.gen!I family, C2LOP.P family, and C2LOP.gen!g family virtually. As shown in Figure 17, the classification performance of the method proposed by Cui et al. [4] on the four families is not good. The VGG16 (fine-tune) method [28] achieved a good classification performance on the C2LOP.P and C2LOP.gen!g families, but the classification accuracy of the samples in the Swizzor.gen!E family and Swizzor.gen!I family still could not reach 80%. It still fails to solve the problem of sample confusion between these two families.

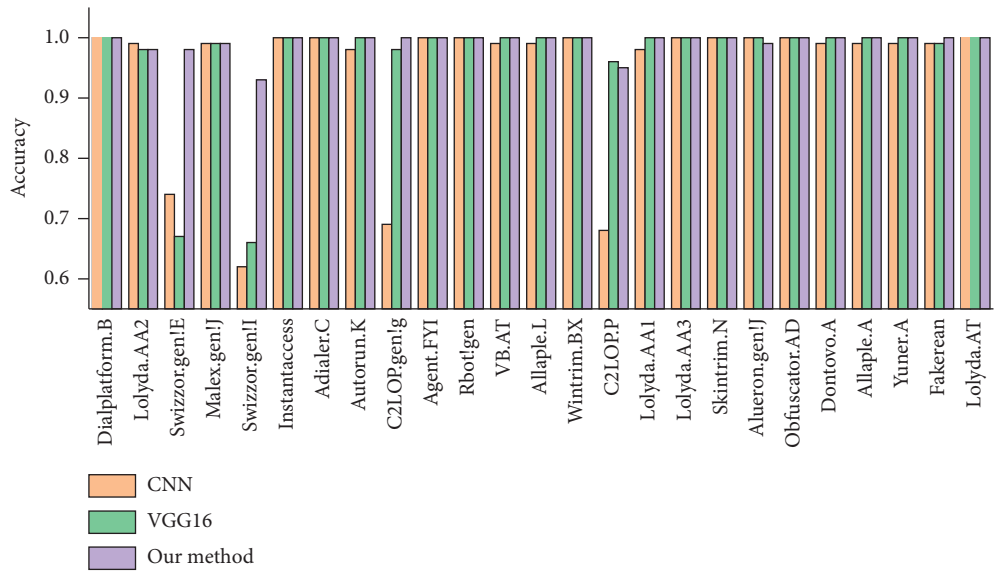


Figure 17: Comparison with other methods on each malware family.

However, the proposed method has achieved good classification performance in all these four families, and the classification accuracy of each family could reach 90%. The experimental results further prove that the proposed method has a better classification performance than other methods, especially for some similar family samples, which can achieve higher classification accuracy.

5. Limitation and Discussion

In this paper, a ten-fold cross-validation experiment is performed on the experimental dataset. From the experimental results, the proposed method can achieve better malware classification performance. There are two main reasons why our method can achieve better classification performance. The first reason is that we use different deep networks. After a large number of experiments, these new types of deep learning networks can achieve better performance based on these three types of features. The second reason is that we have proposed new malicious code features. Based on the related research, we adopted a variety of malicious code data representation methods and extracted a variety of features, including RGB and entropy features. These features can more accurately describe the malicious code to help achieve a better classification performance. For the entropy features, the entropy visualization method proposed in this paper could improve the final classification performance. However, the classification performance of the entropy feature classifier still needs to be improved. In future work, it is necessary to apply more methods to achieve higher classification performance. Besides, the proposed method extracts the features mainly based on the raw byte of malicious code, and it still has some limitations. In future research, the proposed method can be used combined with different kinds of methods to solve various problems encountered in practical applications.

The combination of these three types of features effectively improves the performance of malicious code classification, and the selection of different deep learning networks further improves the classification performance. However, how to quickly find the relatively optimal deep learning network according to the characteristics of the dataset is indeed worthy of in-depth study. In future work, we will conduct further research on the settings of the deep network and image format to propose new malicious code classification methods.

6. Conclusions and Future Work

This paper proposed a new malware classification method based on multiple visualization features and the transfer algorithm. First, the raw byte of malware is converted into byte sequence RGB image, location information RGB image, and entropy grayscale image, respectively. Then, three types of features are extracted based on different deep learning networks. Finally, the three types of features are combined to train a classifier. We implement the method and evaluate it on two custom datasets with a total of more than 20,000 samples provided by the Malware Research Lab and Microsoft Research. The experimental results show that the proposed method can effectively facilitate us in distinguishing malware variants of similar families, and the values of classification accuracy on the two datasets all could reach 99.5%, showing its superiority over other methods. Moreover, this paper also proves that we could solve the misclassification problem of malware samples from similar families based on visual features. The visual characteristics of malware are of great significance to further improve the performance of malware classification in the future.

In future work, we would further explore the entropy feature extraction method and propose better malicious code classification methods. Besides, with the rapid development of deep learning algorithms and natural language processing algorithms, we would conduct in-depth research

on how to improve the classification performance based on new algorithms.

Data Availability

The malware samples used to support the finding of this study are available at DOI: 10.1145/2016904.2016908 and BIG 2015 (available at https://ww.kaggle.com/c/malware-classification). These prior studies (and datasets) are cited at relevant places within the text as references [3, 43].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors are very thankful to Zhi-hao Yan, Zhi-jie Xie, Xuan-zhen Guo, Hang Zhou, Rui-peng Wang, and Yuan-chao Chen for their help in the preparation of experiment and paper review. This research was funded by the Laboratory of Network Security, College of Electronic Engineering, National University of Defense Technology, Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China, and the Natural Science Foundation of Anhui Provincial (grant number 1908085QF291).

References

[1] SYMANTEC, "Internet security threat report," May 2020, https://www.broadcom.com/support/security-center/publications/threat-report.

[2] D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," *Computers & Security*, vol. 81, pp. 123–147, 2019.

[3] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath, "Malware images: visualization and automatic classification," in *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, pp. 1–7, Pittsburgh, PA, USA, July 2011.

[4] Z. Cui, F. Xue, X. Cai, Y. Cao, G.-g. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3187–3196, 2018.

[5] L. Nataraj, V. Yegneswaran, P. Porras, and J. Zhang, "A comparative assessment of malware classification using binary texture analysis and dynamic analysis," in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pp. 21–30, Chicago, IL, USA, October 2011.

[6] H. Naeem, B. Guo, M. R. Naeem, F. Ullah, H. Aldabbas, and M. S. Javed, "Identification of malicious code variants based on image visualization," *Computers & Electrical Engineering*, vol. 76, pp. 225–237, 2019.

[7] Z. Tang, P. Wang, and J. Wang, "Convprotonet: deep prototype induction towards better class representation for few-shot malware classification," *Applied Sciences*, vol. 10, no. 8, p. 2847, 2020.

[8] M. Wojnowicz, G. Chisholm, M. Wolff, and X. Zhao, "Wavelet decomposition of software entropy reveals symptoms of malicious code," *Journal of Innovation in Digital Ecosystems*, vol. 3, no. 2, pp. 130–140, 2016.

[9] M. Bat-Erdene, H. Park, H. Li, H. Lee, and M.-S. Choi, "Entropy analysis to classify unknown packing algorithms for malware detection," *International Journal of Information Security*, vol. 16, no. 3, pp. 227–248, 2017.

[10] L. Liu, X. He, L. Liu, L. Qing, Y. Fang, and J. Liu, "Capturing the symptoms of malicious code in electronic documents by file's entropy signal combined with machine learning," *Applied Soft Computing*, vol. 82, Article ID 105598, 2019.

[11] G. Canfora, F. Mercaldo, and C. A. Visaggio, "An hmm and structural entropy based detector for android malware: an empirical study," *Computers & Security*, vol. 61, pp. 1–18, 2016.

[12] S. S. W. Piyanuntcharatsr, S. Adulkasem, and C. Chantrapornchai, "On the comparison of malware detection methods using data mining with two feature sets," *International Journal of Security and Its Applications*, vol. 9, no. 3, pp. 293–318, 2015.

[13] Z. Feng, S. Xiong, D. Cao et al., "A hybrid framework for malware detection," in *Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics*, pp. 19–26, San Antonio, TX, USA, March 2015.

[14] E. Raff and C. Nicholas, "An alternative to ncd for large sequences, lempel-ziv jaccard distance," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1007–1015, Halifax, Canada, August 2017.

[15] D. Kong and G. Yan, "Discriminant malware distance learning on structural information for automated malware classification," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1357–1365, Chicago, IL, USA, August 2013.

[16] J. Upchurch and X. Zhou, "Variant: a malware similarity testing framework," in *Proceedings of the 2015 10th International Conference on Malicious and Unwanted Software (MALWARE)*, pp. 31–39, IEEE, Fajardo, PR, USA, October 2015.

[17] N. Kawaguchi and K. Omote, "Malware function classification using apis in initial behavior," in *Proceedings of the 2015 10th Asia Joint Conference on Information Security*, pp. 138–144, IEEE, Kaohsiung City, Taiwan, May 2015.

[18] P. Vadrevu and R. Perdisci, "Maxs: scaling malware execution with sequential multi-hypothesis testing," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pp. 771–782, Xi'an, China, May-June 2016.

[19] H. Kim, J. Kim, Y. Kim, I. Kim, K. J. Kim, and H. Kim, "Improvement of malware detection and classification using api call sequence alignment and visualization," *Cluster Computing*, vol. 22, no. 1, pp. 921–929, 2019.

[20] Y. Dai, H. Li, Y. Qian, R. Yang, and M. Zheng, "Smash: a malware detection method based on multi-feature ensemble learning," *IEEE Access*, vol. 7, pp. 112 588–112 597, 2019.

[21] C.-T. Lin, N.-J. Wang, H. Xiao, and C. Eckert, "Feature selection and extraction for malware classification," *Journal of Information Science and Engineering*, vol. 31, no. 3, pp. 965–992, 2015.

[22] K. Kosmidis and C. Kalloniatis, "Machine learning and images for malware detection and clxassification," in *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, pp. 1–6, Larissa Greece, September 2017.

[23] H. Naeem, B. Guo, F. Ullah, and M. R. Naeem, "A cross-platform malware variant classification based on image representation," *KSII Transactions on Internet & Information Systems*, vol. 13, no. 7, 2019.

[24] B. Xiaofang, C. Li, H. Weihua, and W. Qu, "Malware variant detection using similarity search over content fingerprint," in *Proceedings of the the 26th Chinese Control and Decision Conference (2014 CCDC)*, pp. 5334–5339, IEEE, Changsha, China, May-June 2014.

[25] H. Hashemi and A. Hamzeh, "Visual malware detection using local malicious pattern," *Journal of Computer Virology and Hacking Techniques*, vol. 15, no. 1, pp. 1–14, 2019.

[26] J. Zhang, Z. Qin, H. Yin, L. Ou, S. Xiao, and Y. Hu, "Malware variant detection using opcode image recognition with small training sets," in *Proceedings of the 2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–9, IEEE, Waikoloa, HI, USA, August 2016.

[27] K. Han, B. Kang, and E. G. Im, "Malware analysis using visualized image matrices," *The Scientific World Journal*, vol. 2014, p. 15, Article ID 132713, 2014.

[28] E. Rezende, G. Ruppert, T. Carvalho, A. Theophilo, F. Ramos, and P. de Geus, "Malicious software classification using vgg16 deep neural network's bottleneck features," in *Information Technology-New Generations*, pp. 51–59, Springer, Berlin, Germany, 2018.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[30] F. Chollet, "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, Honolulu, HI, USA, July 2017.

[31] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[32] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," 2015, https://arxiv.org/abs/1502.03167.

[33] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, Las Vegas, NV, USA, June-July 2016.

[34] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," 2016, https://arxiv.org/abs/1602.07261.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[36] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[37] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," *Diploma, Technische Universität München*, vol. 91, no. 1, 1991.

[38] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1704–1716, 2011.

[39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, Sardinia, Italy, May 2010.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, Santiago, Chile, December 2015.

[41] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5353–5360, Boston, MA, USA, June 2015.

[42] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, Haifa, Israel, June 2010.

[43] Microsoft, "Microsoft malware classification challenge (big 2015)," June 2020, https://www.kaggle.com/c/malware-classification.