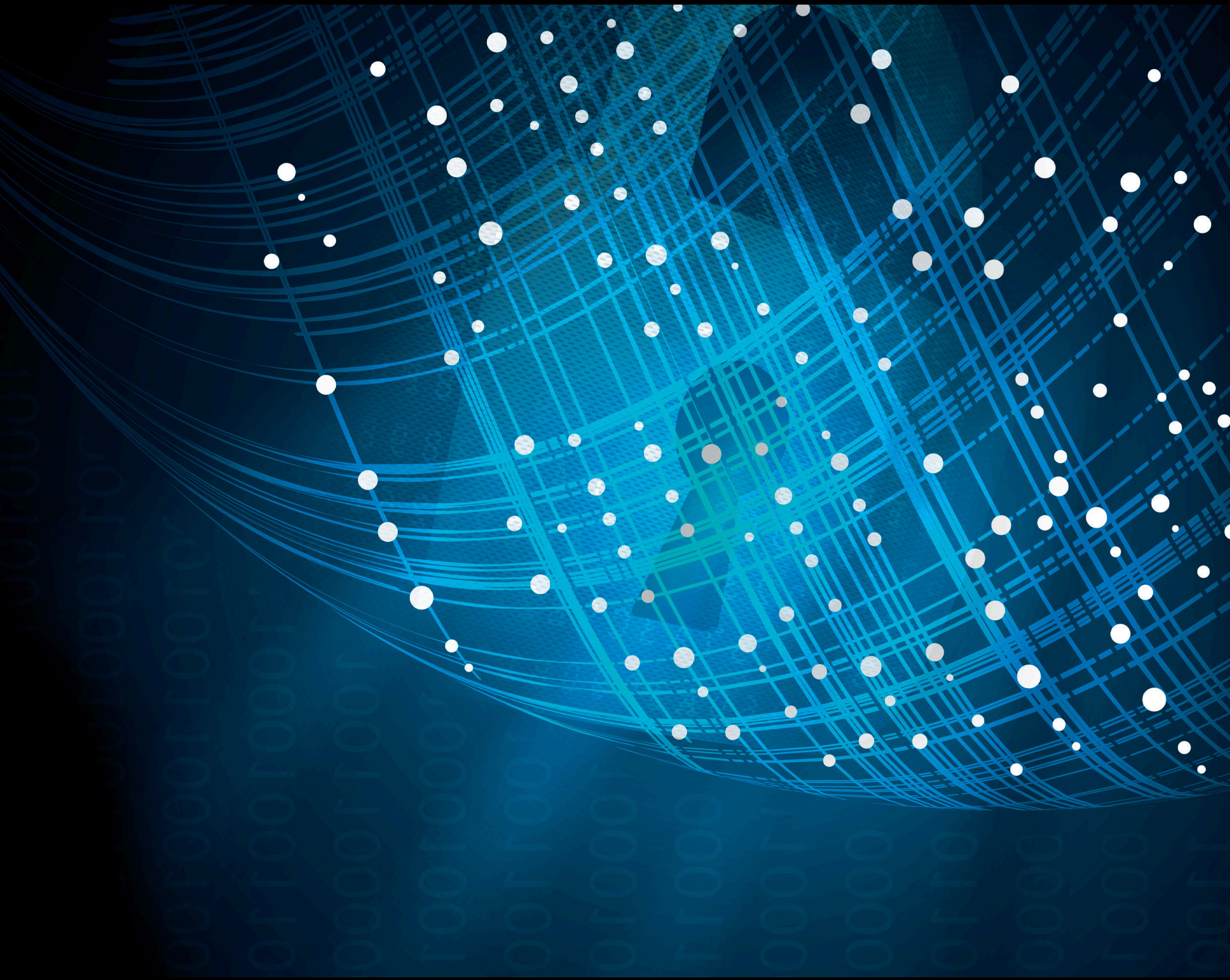
Security and Communication Networks

Security, Trust, and Privacy in Machine Learning-Based Internet of Things

Lead Guest Editor: Weizhi Meng
Guest Editors: Wenjuan Li, Jinguang Han, and Chunhua Su



Security, Trust, and Privacy in Machine Learning-Based Internet of Things

Security and Communication Networks

Security, Trust, and Privacy in Machine Learning-Based Internet of Things

Lead Guest Editor: Weizhi Meng
Guest Editors: Wenjuan Li, Jinguang Han, and Chunhua Su

Copyright © 2022 Hindawi Limited. All rights reserved.

This is a special issue published in "Security and Communication Networks." All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chief Editor

Roberto Di Pietro, Qatar

Editorial Board

Ahmed A. Abd El-Latif, Egypt
Jawad Ahmad, United Kingdom
Mamoun Alazab, Australia
Cristina Alcaraz, Spain
Dr. Jehad Ali, Republic of Korea
Saud Althunibat, Jordan
Ruhul Amin, India
Maria Azees, India
Benjamin Aziz, United Kingdom
Shahram Babaie, Iran
Taimur Bakhshi, United Kingdom
Spiridon Bakiras, Qatar
Pablo Garcia Bringas, Spain
William Buchanan, United Kingdom
Michele Bugliesi, Italy
Jin Wook Byun, Republic of Korea
Pino Caballero-Gil, Spain
Bruno Carpentieri, Italy
Luigi Catuogno, Italy
Shehzad Ashraf Chaudhry, Turkey
Ricardo Chaves, Portugal
Chien-Ming Chen, China
Rongmao Chen, China
Chin-Ling Chen, Taiwan
Tom Chen, United Kingdom
Stelvio Cimato, Italy
Vincenzo Conti, Italy
Luigi Coppolino, Italy
Juhriyansyah Dalle, Indonesia
Salvatore D'Antonio, Italy
Alfredo De Santis, Italy
Angel M. Del Rey, Spain
Roberto Di Pietro, France
Jesús Díaz-Verdejo, Spain
Wenxiu Ding, China
Nicola Dragoni, Denmark
Wei Feng, China
Carmen Fernandez-Gago, Spain
Mohamed Amine Ferrag, Algeria
AnMin Fu, China
Clemente Galdi, Italy
Dimitrios Geneiatakis, Italy
Bela Genge, Romania
Anwar Ghani, Pakistan
Debasis Giri, India
Muhammad A. Gondal, Oman
Prosanta Gope, United Kingdom
Francesco Gringoli, Italy
Biao Han, China
Jinguang Han, China
Weili Han, China
Khizar Hayat, Oman
Jiankun Hu, Australia
Iqtadar Hussain, Qatar
Azeem Irshad, Pakistan
M.A. Jabbar, India
Mian Ahmad Jan, Pakistan
Rutvij Jhaveri, India
Tao Jiang, China
Xuyang Jing, China
Minho Jo, Republic of Korea
Bruce M. Kapron, Canada
Arijit Karati, Taiwan
Marimuthu Karuppiah, India
ASM Kayes, Australia
Habib Ullah Khan, Qatar
Fazlullah Khan, Pakistan
Kiseon Kim, Republic of Korea
Sanjeev Kumar, USA
Hyun Kwon, Republic of Korea
Maryline Laurent, France
Jegatha Deborah Lazarus, India
Wenjuan Li, Hong Kong
Huaizhi Li, USA
Kaitai Liang, United Kingdom
Xueqin Liang, Finland
Zhe Liu, Canada
Guangchi Liu, USA
Flavio Lombardi, Italy
Pascal Lorenz, France
Yang Lu, China
Leandros Maglaras, United Kingdom
Emanuele Maiorana, Italy
Vincente Martin, Spain
Barbara Masucci, Italy
David Megias, Spain
Weizhi Meng, Denmark
Andrea Michienzi, Italy

Laura Mongioi, Italy
Raul Monroy, Mexico
Rebecca Montanari, Italy
Leonardo Mostarda, Italy
HAMAD NAEEM, China, China
Mohamed Nassar, Lebanon
Shah Nazir, Pakistan
Qiang Ni, United Kingdom
Mahmood Niazi, Saudi Arabia
Petros Nicopolitidis, Greece
Vijayakumar Pandi, India
A. Peinado, Spain
Gerardo Pelosi, Italy
Gregorio Martinez Perez, Spain
Pedro Peris-Lopez, Spain
Carla Ràfols, Germany
Francesco Regazzoni, Switzerland
Abdalhossein Rezai, Iran
Helena Rifà-Pous, Spain
Arun Kumar Sangaiah, India
Neetesh Saxena, United Kingdom
Savio Sciancalepore, The Netherlands
Sourav Sen, USA
Young-Ho Seo, Republic of Korea
De Rosal Ignatius Moses Setiadi, Indonesia
Naghmeh Sheykhkanloo, United Kingdom
Wenbo Shi, China
Ghanshyam Singh, South Africa
Daniel Slamanig, Austria
Vasco Soares, Portugal
Salvatore Sorce, Italy
Abdulhamit Subasi, Saudi Arabia
Zhiyuan Tan, United Kingdom
Je Sen Teh, Malaysia
Farhan Ullah, China
Fulvio Valenza, Italy
Sitalakshmi Venkatraman, Australia
Qichun Wang, China
Jinwei Wang, China
Guojun Wang, China
Hu Xiong, China
Xuehu Yan, China
Zheng Yan, China
Anjia Yang, China
Qing Yang, USA
Jiachen Yang, China
Yu Yao, China
Yinghui Ye, China
Kuo-Hui Yeh, Taiwan
Yong Yu, China
Xiaohui Yuan, USA
Sherali Zeadally, USA
Tao Zhang, China
Leo Y. Zhang, Australia
Zhili Zhou, China
Youwen Zhu, China

Contents

Hindawi
Security and Communication Networks
Volume 2022, Article ID 9851463, 3 pages
https://doi.org/10.1155/2022/9851463

Editorial

Security, Trust, and Privacy in Machine Learning-Based Internet of Things

Weizhi Meng,[1] **Wenjuan Li**,[2] **Jinguang Han**,[3] **and Chunhua Su**[4]

[1] *Technical University of Denmark, Kongens Lyngby, Denmark*
[2] *The Hong Kong Polytechnic University, Hong Kong, China*
[3] *Queen's University Belfast, Belfast, UK*
[4] *University of Aizu, Aizuwakamatsu, Japan*

Correspondence should be addressed to Weizhi Meng; weme@dtu.dk

Received 22 April 2022; Accepted 22 April 2022; Published 30 May 2022

Copyright © 2022 Weizhi Meng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Internet of Things (IoT) allows billions of devices in the physical world as well as virtual environments to exchange data with each other intelligently. The worldwide government Internet of Things (IoT) endpoint electronics and communications market will total $21.3 billion in 2022 [1]. For example, smartphones have become an important personal assistant and an indispensable part of people's everyday life and work. However, IoT security has also been a major concern in both academia and industry [2, 3]. The insider threat is one of the major threats to the IoT applications [4], where the attackers can enjoy the resources within the organization or network. For example, Passive Message Fingerprint Attacks (PMFA) [5], a type of insider attacks, can allow several internal nodes to collaborate and compromise a distributed intrusion detection system (DIDS). Hence, there is a need to deploy more suitable security mechanisms to safeguard the IoT and distributed environment, such as traffic filtration [3, 6], trust management [4, 7], and blockchain [8, 9].

Currently, machine learning technique is being widely applied to IoT in order to facilitate performance and efficiency, such as semisupervised learning [10, 11], reinforcement learning [12], and deep learning [13, 14]. For instance, semisupervised learning has been widely studied on how to enhance the detection of spam by leveraging both labeled and unlabeled data [15]. However, machine learning also suffers many issues, which may threaten the security, trust, and privacy of IoT environments. Among these issues, adversarial learning is one major threat, in which attackers may try to fool the learning algorithm with particular training examples and lead to a false result or an inaccurate machine learning model [16, 17].

This Special Issue will focus on cutting-edge research from both the academia and industry and aims to solicit original research and review articles with a particular emphasis on discussing the security, trust, and privacy challenges in machine learning-based IoT. The potential topics focus on the application of machine learning techniques to address security, privacy, and trust issues in IoT systems, networks, and beyond. All submissions have been reviewed by independent reviewers and have undergone several rounds of revisions before being accepted for publication in this Special Issue. After a rigorous review process, a total of 12 papers were finally accepted.

In the first contribution titled "An Unsupervised Learning-Based Network Threat Situation Assessment Model for Internet of Things", Yang et al. [18] presented an unsupervised learning-based network threat situation assessment model that could work in a multisource data IoT network. In the evaluation, they implemented the algorithm with Python and demonstrated that their approach could reach a stronger characterization ability for network threats.

In the second contribution titled "A Key Business Node Identification Model for Internet of Things Security", Xie et al. [19] introduced a key business node identification model for IoT networks, by providing an analysis of business continuity. It contains four major modules: data preparation module, data operation module, decision module, and analysis module. The experimental results indicated that the proposed model can enhance the identification accuracy, with reasonable continuity risk assessment.

In the third contribution titled "A Privacy-Preserving Caching Scheme for Device-to-Device Communications", Zhong et al. [20] introduced a privacy-preserving device-to-device (D2D) caching scheme by defining the node importance as the weighted sum of the physical intimacy and request similarity between devices. In their comparison with Leave Copy Everywhere (LCE) and Most Popular Cache (MPC), the proposed scheme demonstrated better performance.

In the fourth contribution titled "Two-Party Secure Computation for Any Polynomial Function on Ciphertexts under Different Secret Keys", Jiang [21] introduced a scheme that can reduce the size of the ciphertext under a single key. In the fifth contribution titled "An Efficient Anonymous Communication Scheme to Protect the Privacy of the Source Node Location in the Internet of Things", Li et al. [22] introduced an efficient anonymous communication scheme to ensure privacy in two aspects: source node location and the workload.

In the next contribution titled "A Residual Learning-Based Network Intrusion Detection System", Man and Sun [23] designed a deep learning-based intrusion detection model based on residual learning. There are three parts: data preprocessing, model construction, and model evaluation. Their evaluation on UNSW-NB15 demonstrated that the proposed scheme can reach good performance due to the residual blocks.

In the next contribution titled "Machine Learning-Based Stealing Attack of the Temperature Monitoring System for the Energy Internet of Things", Li et al. [24] designed a platform of Energy Internet of Things (EIoT) for the temperature monitoring system. They then introduced a two-step model stealing attack that can use the stolen data to set a copycat network, which could leak the artificial intelligence models.

In the next contribution titled "An Efficient Communication Intrusion Detection Scheme in AMI Combining Feature Dimensionality Reduction and Improved LSTM", Lu and Tian [25] introduced a Stacked Autoencoder method to achieve feature dimensionality reduction for the high-dimensional features of data in Advanced Metering Infrastructure (AMI). In addition to using Attention Mechanism, their evaluation showed that better performance could be achieved based on two datasets: UNSW-NB15 and NSL-KDD.

In the next contribution titled "An Adaptive Communication-Efficient Federated Learning to Resist Gradient-Based Reconstruction Attacks", Li et al. [26] introduced an adaptive frequency-compression federated learning (AFC-FL) by adjusting the communication frequency and parameter compression. In the evaluation, they showed that the proposed model could reduce the workload significantly.

In the next contribution titled "A Hierarchical Approach for Advanced Persistent Threat Detection with Attention-Based Graph Neural Networks", Li et al. [27] introduced a hierarchical approach that is capable of effectively detecting APTs with attention-based Graph Neural Networks (GNNs). In the evaluation, they discussed that the proposed method could outperform some similar approaches.

In the next contribution titled "Towards a Statistical Model Checking Method for Safety-Critical Cyber-Physical System Verification", Xie et al. [28] constructed a cross-entropy optimization model in Safety-Critical Cyber-Physical System (SCCPS). Their experimental results indicated that the proposed method could reduce the standard deviation and corresponding errors by more than an order of magnitude.

In the final contribution titled "Cost-Sensitive Approach to Improve the HTTP Traffic Detection Performance on Imbalanced Data", Li et al. [29] introduced a character-level abstract feature extraction approach (cost-effective) to enhance the detection of the HTTP traffic under imbalanced data. In the evaluation, they demonstrated a higher detection rate as compared with two similar studies.

Conflicts of Interest

We declare no conflicts of interest.

Acknowledgments

We would like to take this opportunity to thank the Chief Editor Dr. Di Pietro and all staff from *Security and Communication Networks*, for supporting and guiding this Special Issue. We also thank all authors for their submissions and support and express our deepest gratitude to all the anonymous reviewers who devoted their time reviewing all the papers in this Special Issue. We believe that this Special Issue can provide useful hints on how to address security, privacy, and trust issues in machine learning-based IoT environments.

Weizhi Meng
Wenjuan Li
Jinguang Han
Chunhua Su

References

[1] Gartner report:, 2021, https://www.gartner.com/en/newsroom/press-releases/2021-06-30-gartner-global-government-iot-revenue-for-endpoint-electronics-and-communications-to-total-us-dollars-21-billion-in-2022.

[2] W. Li, W. Meng, and M. H. Au, "Enhancing Collaborative Intrusion Detection via Disagreement-Based Semi-Supervised Learning in IoT Environments," *Journal of Network and Computer Applications*, vol. 161, pp. 1–9, 2020.

[3] W. Meng, W. Li, and L. F. Kwok, "Towards Effective Trust-Based Packet Filtering in Collaborative Network Environments," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 233–245, 2017.

[4] W. Meng, K.-K. R. Choo, S. Furnell, A. V. Vasilakos, and C. W. Probst, "Towards Bayesian-Based Trust Management for Insider Attacks in Healthcare Software-Defined Networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 761–773, 2018.

[5] W. Li and W. Meng, “PMFA: Toward Passive Message Fingerprint Attacks on Challenge-Based Collaborative Intrusion Detection Networks,” in *Proceedings of the 10th International Conference on Network and System Security (NSS 2016)*, pp. 433–449, Taipei, Taiwan, September 2016.

[6] W. Meng, “Intrusion Detection in the Era of IoT: Building Trust via Traffic Filtering and Sampling,” *Computer*, vol. 51, no. 7, pp. 36–43, July 2018.

[7] A. Rezapour and W.-G. Tzeng, “A Robust Intrusion Detection Network Using Thresholdless Trust Management System with Incentive Design,” *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol. 255, no. 2, pp. 139–154, 2018.

[8] H. Liang, J. Wu, S. Mumtaz, J. Li, X. Lin, and M. Wen, “MBID: Micro-Blockchain-Based Geographical Dynamic Intrusion Detection for V2X,” *IEEE Communications Magazine*, vol. 57, no. 10, pp. 77–83, 2019.

[9] O. Alkadi, N. Moustafa, and B. Turnbull, “A Review of Intrusion Detection and Blockchain Applications in the Cloud: Approaches, Challenges and Solutions,” *IEEE Access*, vol. 8, pp. 104893–104917, 2020.

[10] R. Švihrová and C. Lettner, “A Semi-Supervised Approach for Network Intrusion Detection,” in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, vol. 93, pp. 1–6, Ireland, August 2020.

[11] Y. Zong and G. Huang, “Application of Artificial Fish Swarm Optimization Semi-Supervised Kernel Fuzzy Clustering Algorithm in Network Intrusion,” *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 2, pp. 1619–1626, 2020.

[12] Y. Jiang, K. Zhang, Y. Qian, and L. Zhou, “Reinforcement Learning-Based Query Optimization in Differentially Private IoT Data Publishing,” *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11163–11176, 2021.

[13] C. Wu and W. Li, “Enhancing Intrusion Detection with Feature Selection and Neural Network,” *International Journal of Intelligent Systems*, vol. 36, no. 7, pp. 3087–3105, 2021.

[14] N. Gupta, V. Jindal, and P. Bedi, “CSE-IDS: Using Cost-Sensitive Deep learning and Ensemble Algorithms to Handle Class Imbalance in Network-Based Intrusion Detection Systems,” *Computers & Security*, vol. 112, Article ID 102499, 2022.

[15] S. Hershkop and S. J. Stolfo, “Combining email Models for False Positive Reduction,” in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining-KDD ‘05*, pp. 98–107, Chicago IL USA, August 2005.

[16] A. Abusnaina, A. Khormali, H. Alasmary, J. Park, A. Anwar, and A. Mohaisen, “Adversarial Learning Attacks on Graph-Based IoT Malware Detection Systems,” *ICDCS*, in *Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems*, pp. 1296–1305, Dallas, TX, USA, July 2019.

[17] A. Singh and B. Sikdar, “Adversarial Attack and Defence Strategies for Deep-Learning-Based IoT Device Classification Techniques,” *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2602–2613, 2022.

[18] H. Yang, R. Zeng, F. Wang, G. Xu, and J. Zhang, “An Unsupervised Learning-Based Network Threat Situation Assessment Model for Internet of Things,” *Security and Communication Networks*, vol. 2020, Article ID 6656066, 11 pages, 2020.

[19] L. Xie, H. Ni, H. Yang, and J. Zhang, “A Key Business Node Identification Model for Internet of Things Security,” *Security and Communication Networks*, vol. 2020, pp. 1–11, Article ID 6654283, 2020.

[20] Y. Zhong, Z. Li, and L. Liao, “A Privacy-Preserving Caching Scheme for Device-to-Device Communications,” *Security and Communication Networks*, vol. 2021, Article ID 6696149, 8 pages, 2021.

[21] B. Jiang, “Two-Party Secure Computation for Any Polynomial Function on Ciphertexts under Different Secret Keys,” *Security and Communication Networks*, vol. 2021, pp. 1–6695304, 2021.

[22] F. Li, P. Ren, G. Yang et al., “An Efficient Anonymous Communication Scheme to Protect the Privacy of the Source Node Location in the Internet of Things,” *Security and Communication Networks*, vol. 2021, Article ID 6670847, 16 pages, 2021.

[23] J. Man and G. Sun, “A Residual Learning-Based Network Intrusion Detection System,” *Security and Communication Networks*, vol. 2021, Article ID 5593435, 9 pages, 2021.

[24] Q. Li, L. Zhang, R. Zhou, Y. Xia, W. Gao, and Y. Tai, “Machine Learning-Based Stealing Attack of the Temperature Monitoring System for the Energy Internet of Things,” *Security and Communication Networks*, vol. 2021, Article ID 6661954, 8 pages, 2021.

[25] G. Lu and X. Tian, “An Efficient Communication Intrusion Detection Scheme in AMI Combining Feature Dimensionality Reduction and Improved LSTM,” *Security and Communication Networks*, vol. 2021, Article ID 6631075, 21 pages, 2021.

[26] Y. Li, Y. Li, H. Xu, and S. Ren, “An Adaptive Communication-Efficient Federated Learning to Resist Gradient-Based Reconstruction Attacks,” *Security and Communication Networks*, vol. 2021, Article ID 9919030, 16 pages, 2021.

[27] Z. Li, X. Cheng, L. Sun, J. Zhang, and B. Chen, “A Hierarchical Approach for Advanced Persistent Threat Detection with Attention-Based Graph Neural Networks,” *Security and Communication Networks*, vol. 2021, Article ID 9961342, 14 pages, 2021.

[28] J. Xie, W. Tan, B. Fang, and Z. Huang, “Towards a Statistical Model Checking Method for Safety-Critical Cyber-Physical System Verification,” *Security and Communication Networks*, vol. 2021, Article ID 5536722, 12 pages, 2021.

[29] W. Li, S. Sun, S. Zhang, H. Zhang, and Y. Shi, “Cost-Sensitive Approach to Improve the HTTP Traffic Detection Performance on Imbalanced Data,” *Security and Communication Networks*, vol. 2021, Article ID 6674325, 11 pages, 2021.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6674325, 11 pages
https://doi.org/10.1155/2021/6674325

Research Article

Cost-Sensitive Approach to Improve the HTTP Traffic Detection Performance on Imbalanced Data

Wenmin Li, Sanqi Sun, Shuo Zhang, Hua Zhang, and Yijie Shi

The State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Shuo Zhang; shuozhang@bupt.edu.cn

Received 18 December 2020; Revised 28 April 2021; Accepted 13 May 2021; Published 24 May 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Wenmin Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aim. The purpose of this study is how to better detect attack traffic in imbalance datasets. The deep learning technology has played an important role in detecting malicious network traffic in recent years. However, it suffers serious imbalance distribution of data if the traffic model skews towards the modeling in the benign direction, because only a small portion of traffic is malicious, while most network traffic is benign. That is the reason why the authors wrote this manuscript. *Methods*. We propose a cost-sensitive approach to improve the HTTP traffic detection performance with imbalanced data and also present a character-level abstract feature extraction approach that can provide features with clear decision boundaries in addition. Finally, we design a spark-based HTTP traffic detection system based on these two approaches. *Results*. The methods proposed in this paper work well in imbalanced datasets. Compared to other methods, the experiment results indicate that our system has F1-score in a high precision. *Conclusion*. For imbalanced HTTP traffic detection, we confirmed that the method of feature extraction and the cost function is very effective. In the future, we may focus on how to use the cost function to further improve detection performance.

1. Introduction

1.1. Background. In the past few years, cybersecurity incidents have occurred frequently. In the first half of 2018, 360 Internet Security Center intercepted 140 million malicious programs in all, nearly 795,000 ones per day on average [1]. Moreover, around 8% of Hypertext Transfer Protocol (HTTP) messages in 2017 were reported to be malicious [2].

Deep learning, as one of the most currently remarkable machine learning techniques, has achieved great success in many applications such as image analysis, speech recognition, and text understanding [3]. In the field of objection detection, Girshick et al. [4] greatly improved the accuracy of objection detection through the deep learning technology. Wu et al. [5] used weakly supervised learning to classify and annotate images. Rattani et al. [6] applied deep learning technology to the field of selfie biometrics and has made good progress. In medical image segmentation, U-NET [7] is undoubtedly one of the most successful methods, which was proposed at the MICCAI conference in 2015. In the field of HTTP traffic detection, the deep learning technology is prominent way to detect malicious network traffic. However, it suffers serious imbalanced distribution of data. For example, the traffic detection tasks usually focus on reducing malicious traffic such as web attack, but not the data of web browsing accounts for the majority. The contribution of the majority class to the cost function far exceeds that of the minority class. Therefore, it is difficult to identify the small amount of traffic, which brings serious challenges to network traffic classification [8].

1.2. Related Work. The detection technologies of imbalanced data can be classified into three types: data-level methods, feature extraction, and cost-sensitive learning. Oversampling, undersampling, and random sampling are the most commonly used in data level. Jin et al. [9] and Lim et al. [10] applied the data-level methods to rebalance traffic data and improve the performance of imbalanced dataset detection. Oversampling improves classification performance by increasing the number of the minority class samples.

However, due to the large number of copies of the minority class samples, the classification algorithm is difficult to avoid overfitting. Undersampling improves classification performance by reducing the number of the majority class samples. However, in the field of HTTP traffic detection, the majority class samples are far more than the minority class samples, and the quantity difference may be hundreds of times, so the downsampling method may not be suitable. Random sampling randomly abandons the minority class samples, which may remove potentially useful information from the minority class samples. Park et al. [11] proposed an anomaly detection technique for imbalanced HTTP traffic utilizing convolutional autoencoders (CAE), which belongs to the type of feature extraction. However, converting HTTP massage into an image via one-hot encoding will lose some original information. And we will improve the feature extraction method mentioned in the paper. Another common method is cost-sensitive learning, which uses a cost function to train the classifier. The cost-sensitive method in [12] is based on decision tree. However, due to the complexity of the problem, the performance of the algorithm based on neural network is usually better than the algorithm based on decision tree in the field of HTTP traffic detection. Chen et al. [13] introduced a novel imbalanced classification model, named simplex imbalanced data gravitation classification (S-IDGC). This model uses Euclidean distance to calculate gravity, but fails to consider the data distribution characteristics and the results were in a low precision. Recently, the focal-loss cost function proposed by Lin et al. [14] has been proved to be effective in the field of image segmentation. This method performs well in image segmentation. Tong [15] proposed a traffic classification method based on convolutional neural network which consists of two main traffic classification stages and combines the flow and packet-based features to predict the services based on quick UDP internet connection. Aceto et al. [16] used multimodal deep learning to study mobile encrypted traffic classification which has a good result about TSL traffic detection. Lotfollahi et al. [17] combined port-based, payload inspection and statistical machine learning to analyze encrypted traffic classification. Bovenzi [18] imposed a hierarchical hybrid intrusion detection approach, which has been proved to be very effective in the Internet of things scenario. Aceto [19] firstly investigated and experimentally evaluated the adoption of DL-based network traffic classification strategies as supported by BD frameworks. The recent schemes focused on the mobile or light equipment and they analyzed encrypted traffic classification. However, in the field of HTTP traffic detection, the contribution of the minority class samples to the loss function will be reduced according to the predicted value in the training model, which is not conducive to the detection of the minority class samples. The characteristics of recent related works are shown in Table 1.

Contributions: the main motivation of this paper is to detect attacks from serious imbalanced network traffic. To achieve this goal, we address it from two aspects: feature extraction and cost function. The main contributions of this paper can be summarized as follows:

(i) In terms of feature extraction, we present character-level abstract feature extraction approach which can provide features with clear decision boundaries.

(ii) In terms of cost function, we present the HM-loss cost function to improve the http traffic detection performance on imbalanced data. The cost-sensitive approach can reduce the contribution of the majority class in the cost function.

(iii) Finally, we design and implement spark-based HTTP traffic detection system and apply the cost-sensitive approach and the feature extraction approach into this detection system. The experiment results show that proposed scheme has higher precisions, recall, and F1-score.

The rest of this paper is organized as follows. Section 2 is a detailed description on character-level abstract feature extraction approach whereas Section 3 describes cost-sensitive approach. The experiment is given in Section 4. Conclusion and future directions are given at the end of the paper.

2. Methods

2.1. The Character-Level Abstract Feature Extraction Approach. We present the character-level abstract feature extraction approach in this section which combines character-level features and abstract features. Our main work is to extract character-level features based on spark [20] clusters, design a one-dimensional convolutional autoencoder, and then extract abstract features.

For the feature extraction of http traffic, n-gram feature [21] and character-level feature [22] are the most popular methods for converting HTTP messages into input vectors fed into neural networks. However, n-gram features can cause a large amount of information loss and have higher feature dimensions, and character-level features have no clean decision boundaries on imbalanced data because it contains a lot of noise information. Considering that the abstract features generated by CAE have clean decision boundaries, but CAE cannot directly obtain input vectors from http traffic, we present the character-level abstract feature extraction approach based on character-level features. The workflow of the abstract feature extraction approach is shown in Figure 1.

2.2. Character-Level Feature Extraction Method Based on Spark. Zhang et al. [22] have done a lot of research on character-level features. However, Zhang's experiment is implemented on a single computer. In an actual production environment, it will encounter a calculation bottleneck. Therefore, we extend the character-level feature extraction method on spark and combine it with abstract features to extract character-level abstract features used in the paper.

We perform preprocessing steps and extract character-level features on the spark cluster. First, we install and configure the Hadoop cluster [23] on the ubuntu server, and our spark mode is spark on yarn. Second, we allocate appropriate computing resources for spark tasks based on the amount of task data and expected completion time. For example, the gateway produces 15 GB http traffic stored on Hadoop Distributed File System (HDFS) every 5 minutes. We set the size of HDFS default block to 128 MB, so the traffic can be split into 120 (15 GB/128 MB) tasks. Then, we assume that 120 tasks need to run 2 to 3 times in the cluster (according to experience, this configuration can maximize resource utilization). Therefore, we can assign 50 executor instances to the cluster, each instance assigning 2 to 3 CPU. Third, we write spark programs with the Jupyter notebook tool. The specific steps of data preprocessing and feature extraction are same as in the paper [22]. The pseudocode of the character-level feature extraction algorithm is shown in Algorithm 1. In pseudocode, we will merge the URL and post fields into feature string and then filter out non-ASCII characters. In the end, we need to generate a string of fixed length L, and if the length is greater than L, truncate it; if the length is less than L, repeat filling until the length reaches L. After extracting the character-level features, we transfer the feature data into Kafka [24] system for using in subsequent steps.

For malicious http traffic, the URL and body fields are more likely to contain sensitive attack information. Therefore, this paper chooses these two fields as the main detection target. Part of the training set containing only the URL field and the body field is shown in Table 2.

2.3. Abstract Feature Extraction by Autoencoder. The section mainly presents the workflow of extracting the abstract feature generated by the one-dimensional CAE. The main motivation of the abstract features generated by CAE is to generate a clean decision boundary. Therefore, we use the one-dimensional CAE to generate abstract features by learning the character-level feature. The results show that this method can effectively reduce the impact of imbalanced data distribution on the malicious traffic detection.

In Figure 2, the classic CAE's input is two-dimensional images. But the URL and post fields for HTTP traffic are one-dimensional, unlike images with two-dimensional spatial information; the paper processes the input text data into one-dimensional.

Figure 3 shows the structure of the one-dimensional CAE designed by us. Each layer of the encoder consists of multiple nodes, and the last hidden layer of the encoder generates the abstract feature. The decoder also has a multilayered structure that is symmetrical with the corresponding layer in the encoder, and the last layer of decoder is the output layer. The cost function calculates the error based on the output layer and the input layer. And, to reduce overfitting, the dropout ratio between the encoder and the decoder is set to 0.1.

CAE is unsupervised learning, so manual labels are not required and the CAE's input is $300 * 1$ character, and after convolution steps, an abstract feature of size $25 * 8$ is generated.

3. The Cost-Sensitive Approach

We present the HM-loss cost function in this section. Our main work consists of two parts. First, we describe the disadvantages of the CE-loss when dealing with imbalanced http traffic. Second, we design the HM-loss cost function which is cost-sensitive. In this approach, we design a coefficient for the loss function and when the classification algorithm predicts the minority class samples, the weight coefficient factor can dynamically adjust the contribution of the majority class samples to the loss function. When the minority class samples are predicted, the contribution of the sample of the loss function is kept unchanged.

3.1. The Disadvantages of the CE-Loss on Imbalanced HTTP Traffic. The CE-loss function used as the cost function for deep learning techniques is very popular in the classification task. However, it suffers from a low F1-score value when dealing with severely imbalanced HTTP traffic in an actual production environment. Because the contribution of the majority class to the cost function far exceeds that of the minority class, the model's decision tends to support the majority class and ignore the minority exception class [14].

Figure 4 shows the loss value of the CE-loss varies with the prediction probability. As shown in the picture, the predicted value of a single normal sample tends to be large, close to 1, but the contribution to the loss function is small. Conversely, the predicted value of a single malicious sample tends to be small (less than the normal sample), but contributes a lot to the loss function.

Now, we assume that the predicted probability of a normal sample is 0.97, and we can calculate that the sum of contribution to the cost function of 500,000 normal samples is 15,229. At the same time, assuming that the predicted probability of a malicious sample is 0.88, the sum of contribution to the cost function of 705 malicious samples is 39.14. The loss value of the benign sample was about 389 (15,229/39.14) times that of the malicious sample. Therefore, in the backpropagation of the neural network, the loss of normal samples dominates the decline of the gradient, and the algorithm focuses on the majority class.

3.2. The Definition of HM-Loss Cost Function. In this part, we design the HM-loss cost function and give the definition of HM-loss cost function. First, we present the idea of HM-loss, then give the definition of HM-loss, and finally summarize the advantages and characteristics of HM-loss.

The main idea of the HM-loss cost function is to dynamically adjust the weight of sample's contribution to the loss value [14]. When the true label belongs to the majority category (negative), the weight of the contribution to the loss decreases, and the degree of decrease varies according to the predicted probability value, and usually, when the prediction is correct, the contribution to the loss decreases greatly. In addition, our cost function has another property. When the true label belongs to the minority class (positive), the weight of contribution to the loss remains. Therefore, we can adjust the algorithm to focus on the minority class samples by giving the majority class samples less attention [25]. We focus on the minority class samples but not the majority class samples.

The idea of the HM-loss cost function is present in the previous paragraph. Now, we give the specific definition. The definition of HM-loss is shown in formula (1). "$ytrue$" represents the real label, and there are only two values of 0 and 1, where 0 represents the positive class and 1 represents the negative class. "$ypred$" represents the prediction probability value, which ranges from 0 to 1.

$$\text{loss} = -\left(ytrue * \cos\left(\alpha * ypred\right)^{\gamma} + \left(1 - ytrue\right)\right) * \log\left(ypred\right), \quad ytrue\, \varepsilon\{1, 0\} \text{ and } \alpha\varepsilon\left(0, \frac{\pi}{2}\right). \tag{1}$$

The HM-loss cost function derived from the CE-loss consists of two parts. The first part is "$ytrue * \cos(\alpha * ypred)^{\gamma}$" which controls the weight that varies from ypred value. The two hyperparameters contained in this part are to adjust the degree of weight reduction. The second part is "$(1 - ytrue)$" which controls the weight of the minority class's contribution to the loss function and it remains unchanged. Figure 5 shows the loss value of the HM-loss cost function under different hyperparameters.

We explain the definition of the HM-loss cost function in the previous section, and we present the advantages of it as follows. The first advantage of the HM-loss cost function is that the contribution of the majority class samples to the loss function can be dynamically reduced according to the predicted value. The second is that, regardless of whether the minority classes samples are predicted correctly or not, its contribution to the loss function does not change. The third is that only when the majority class samples are correctly predicted and the probability value is close to 1, the weight value decreases faster.

Now, we take a simple example of what the HM-loss cost function does. Figure 6 shows the loss of 500,000 normal samples under different loss functions, and to show the data better, we select the data with the probability value between 0.85 and 1. We assume that the probability of the majority class sample is 0.97, and we can calculate that the sum of 500,000 samples loss value under the CE-loss cost function is 15,229. Similarly, we can calculate that the loss using the HM-loss cost function under different hyperparameters is 4642, 431, and 40, respectively. It can be seen that the HM-loss cost function is very effective in this case.

4. Results

4.1. Experiment. In this section, we explain the details of the datasets and the performance metrics used in the experiments. Using these metrics, we compare the performance of the proposed scheme with related schemes including data-level methods and Park's method [11]. The experiment consists of three parts. Firstly, we introduce the preparation stage of the experiment, including the dataset and the experimental evaluation index. Then, we show the structure of convolutional neural network used to detect malicious samples. The third part shows the experimental results.

4.2. Experiment Setup. We use real traffic data accumulated over time to validate our approach. And we collect around 701,000 HTTP messages from the gateway of a university in 2019 for this experiment. The collected data is highly sensitive because it contains most of the network activities during work hours of teachers and students. For these data, we perform manual verification and tagging. The types and quantities of malicious samples are shown in Table 3. The numbers of normal and anomalous HTTP messages are around 700,000 and 1,000, respectively. We divide it into training datasets and test datasets according to a certain proportion.

Existing studies have shown that AUC has certain limitations in performance evaluation, especially when the numbers of normal and anomalous messages are significantly different [26, 27]. Therefore, we use F-score [28]. Specifically, the F-score directly related to the recall and the precision is the harmonic average of the precision and recall, and the specific definition is shown in the following formula.

$$F = \frac{2}{(1/\text{recall}) + (1/\text{precision})}. \tag{2}$$

4.3. The Neural Network Model Structure. After obtaining character-level abstract features, we can train the CNN network to classify samples. Our model is based on one-dimensional convolutional neural network which can acquire more local feature. The reason for using one-dimensional vector is that HTTP traffic has no two-dimensional space attributes. The structure of the model used by this experiment mainly includes the input layer, the hidden layer, and the output layer. The input layer first converts the input data into the input tensor fed into one-dimensional convolutional neural network. After the convolution, pooling, ReLU, and flattening steps, the softmax function produces the prediction value. The model architecture is shown in Figure 7.

Table 1: Comparison of related works.

Scheme	Method	Encrypted or not	Friendly to lightweight devices	Friendly to a few samples
Jin et al. [9]	Data-level method	No	No	No
Lim et al. [10]	Data-level method	No	No	No
Park et al. [11]	Convolutional autoencoders	No	No	No
Ting [12]	Decision tree	No	No	No
Chen et al. [13]	Implex imbalanced data gravitation classification	No	No	No
Lin et al. [14]	The focal-loss cost function	No	No	No
Tong [15]	Convolutional neural network	No	No	No
Aceto et al. [16]	Multimodal deep learning	Yes	Yes	No
Lotfollahi [17]	Port-based, payload inspection and statistical machine learning	Yes	No	No
Bovenzi [18]	Hierarchical hybrid intrusion detection	Yes	Yes	No
Aceto [19]	Big data-enabled DL framework for mobile TC	Yes	Yes	No
Ours	Spark-based HTTP traffic detection	No	Yes	Yes

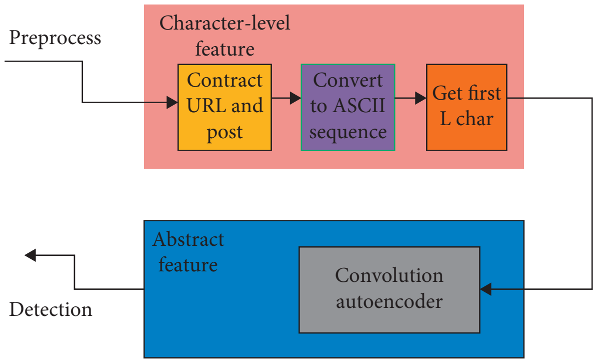


Figure 1: Feature extraction flow chart.

```
Input: HTTP traffic path
Configure the resources occupied by the spark task
Init spark session
Initialize: Truncated fixed length: $L$, result: res
feat-contract URL and post
Filter non-ASCII characters of feat
if(the length of feat $\geq L$){
    feat = the first $L$ character of the feat
}
else{
      do{
          feat = merge two feat strings
      }
      While(getLength( feat) > $L$)
}
if(the length of feat $\geq$ L){
      feat = the first $L$ character of the feat
}
Return feat; //return the string of fixed length
```

Algorithm 1: Character-level feature (HTTP traffic path).

Table 2: URL and post features.

Label	Post	URL
1	−7 = @eval(get_magic_quotes_gpc()?stripslashes($...	http://weki.php/admin/login.action
1	sqzr = @eval(get_magic_quotes_gpc()?stripslashes($...	http://plus/sdfg.php//plusmytag_js.php?aid=8080
1	C = /var/www/vhosts/13/133103/webspace/httpdocs/...	http://wp-includes/js/crop/data.php
1	z2 = ???php+\r**ntitleline = file('key.txt')**;\r\n...	http://upload/2015/09/07/1441610150952000.jsp
1	action = editfile&fname = d:\wwwroot\www.jsjyjxx.c...	http://com4.yichen.asp?action2=post
1	sqzr = response.write("------>/");var err:except...	http://news/pics/20151017/201510171445088639847.php
0	-----------------------a1fa340557\0 × 0d\0...	http://upload.php?hid=sgpy-windows-generic-device-id...
0	api_token = f04362c49325b5cf1ef9d373e4fb89dc16d7...	http://kancolle/proxykancolleapi?h=125.6.189.247&p=/...
0	-----------------------cqdems00sd0wevzsr...	http://cloudquery.php
0	ejx9k8ty4yaqrb/gs1mkeahpmys81vmfruyhmxojoqg5cb...	http://restapi.php

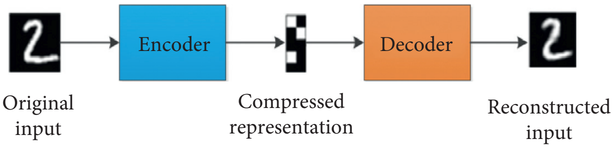


Figure 2: Autoencoder structure.

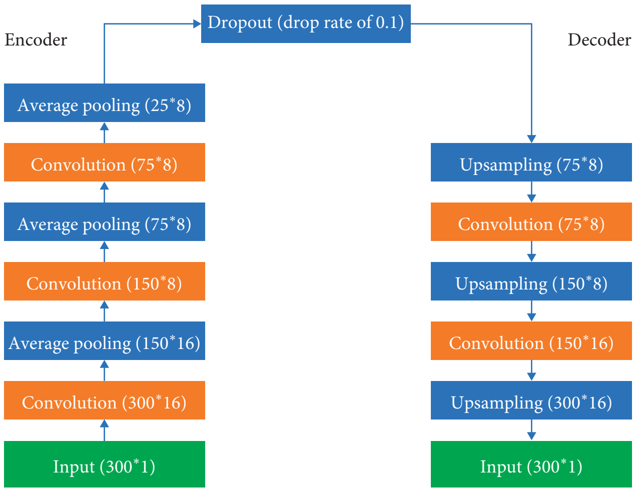


Figure 3: One-dimensional convolution autoencoder architecture.

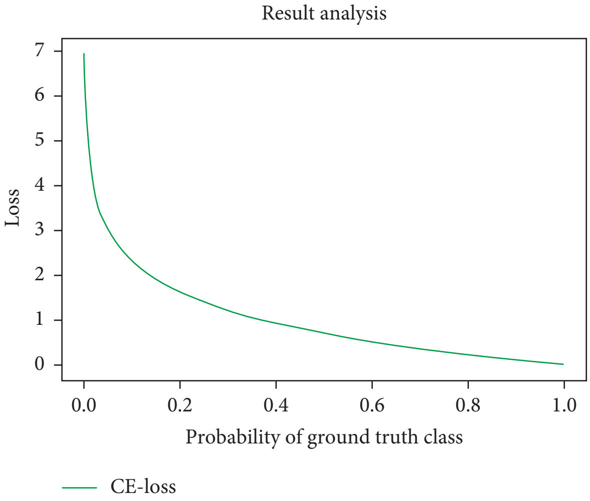


Figure 4: The loss value of CE-loss varies with the prediction probability.

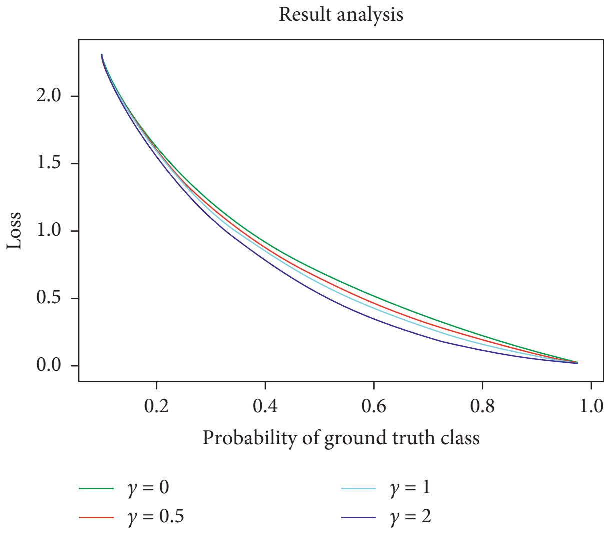


FIGURE 5: $-\cos\ (\alpha * ypred)^{\gamma} *\ \log(ypred)$ with different values in γ.

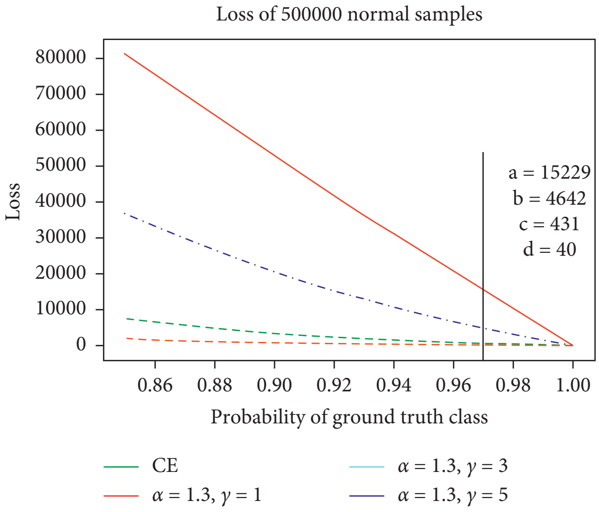


FIGURE 6: Contribution of the 500,000 normal sample to the loss function.

4.4. Experiment Result

4.4.1. HM-Loss Cost Function with Different Hyperparameters. The purpose of this experiment is to find the best two hyperparameters of HM-loss, the experimental data used in this paper.

Table 4 shows that when the hyperparameter alpha and gamma take 1.3 and 3, respectively, the HM-loss performs best on the dataset. And precision recall and F1-score value can reach 0.90, 0.84, and 0.87, respectively.

4.4.2. Comprehensive Experimental Results. The following experimental results are divided into two parts. First, we verify the effectiveness of the character-level feature extraction method. Second, we compare our method with other methods.

(i). We compare Park's feature and character-level abstract features. In this process, we apply different feature extraction methods, but the same algorithm. As shown in Figure 8, our feature has a higher F-score value. Therefore, through this comparative experiment, we can conclude that our feature can work.

(ii). We compare our method with others' methods. The ordinary method which does not adopt any strategy. The oversampling method focuses on the data level. Park's method mentioned in paper [11] focuses on feature extraction. And Lin et al.'s method mentioned in the paper [14] mainly focuses on cost function, which is effective in the field of computer vision. Our method focuses on both feature extraction and cost functions.

Table 3: Dataset.

	Normal	Malicious	Total
Training set	500000	705	500705
Test set	250000	352	250352
total	700000	1057	701057

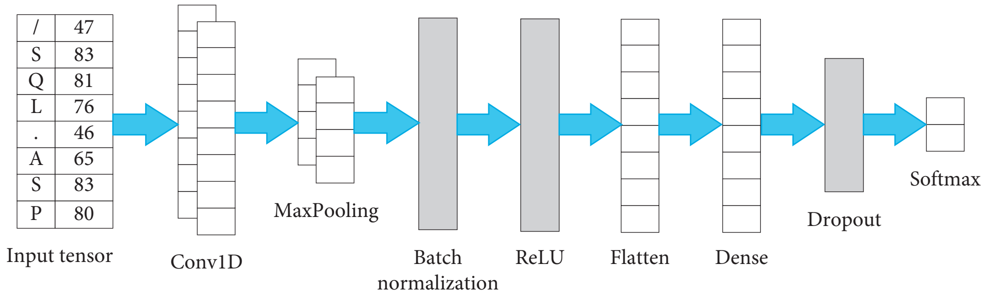


Figure 7: Convolutional neural network structure.

Table 4: The performance metrics when the HM-loss's hyperparameters are assigned different values.

Alpha	Gamma	Precision	Recall	*f*1-score
1.1	1	0.89	0.71	0.79
1.1	3	0.89	0.67	0.76
1.1	5	0.91	0.76	0.83
1.3	1	0.91	0.75	0.82
1.3	3	0.90	0.84	0.87
1.3	5	0.90	0.8	0.85
1.4	1	0.87	0.81	0.84
1.4	3	0.84	0.83	0.83
1.4	5	0.88	0.8	0.84
Pi/2	1	0.86	0.82	0.84
Pi/2	3	0.82	0.79	0.8
Pi/2	5	0.86	0.83	0.84

As shown in Table 5, through the comparative experiment of ordinary method and our method, we can conclude that our method can work. Comparing our method with other method, we can find that our method has higher accuracy and F-score than oversampling technique, Park's method, and Tsung-Yi Lin's method. From the ROC curve comparison results of the three methods in Figure 9, under the same FPR, the HM loss method proposed by us can obtain higher TPR than the method proposed by Tsung-Yi Lin's and Park's, which is better than the other two methods.

According to the experimental results, we can conclude that our method works and performs better than the above related methods when dealing with the imbalanced traffic dataset.

5. Discussion

In this paper, we propose a cost-sensitive approach to improve the HTTP traffic detection performance with imbalanced data. In this approach, we design a coefficient for the loss function and when the classification algorithm predicts the minority class samples, the weight coefficient factor can dynamically adjust the contribution of the majority class samples to the loss function. When the minority class samples are predicted, the contribution of the sample to the loss function is kept unchanged. The experimental results show that this approach is more effective than others. In addition, we also present a character-level abstract feature extraction approach that can provide features with clear decision boundaries in addition. In conclusion, the methods proposed in this paper work well in imbalanced datasets. Compared to other methods, the experiment results indicate that our system has F1-score in a high precision. For imbalanced HTTP traffic detection, we confirmed that the method of feature extraction and cost function is very effective.

In our future work, we will analyze the influence of different types of autoencoders in character-level abstract feature extraction and examine their capabilities and characteristics of improving the performance, whether streamlined autoencoders can keep precision and increase computational efficiency. The theoretical causes of the results require more rigorous regulation. In addition, we will explore more about the performance of stacked autoencoder

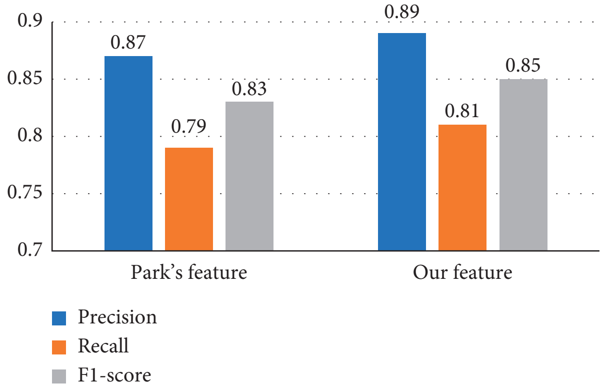


Figure 8: The performance metrics on the ordinary method and our method.

Table 5: The performance metrics on different methods.

	Precision	Recall	*F*1-score
Ordinary method	0.94	0.35	0.51
Oversampling	0.79	0.82	0.81
Park's method	0.87	0.79	0.83
Tsung-Yi Lin's method	0.85	0.83	0.84
Our method	0.9	0.84	0.87

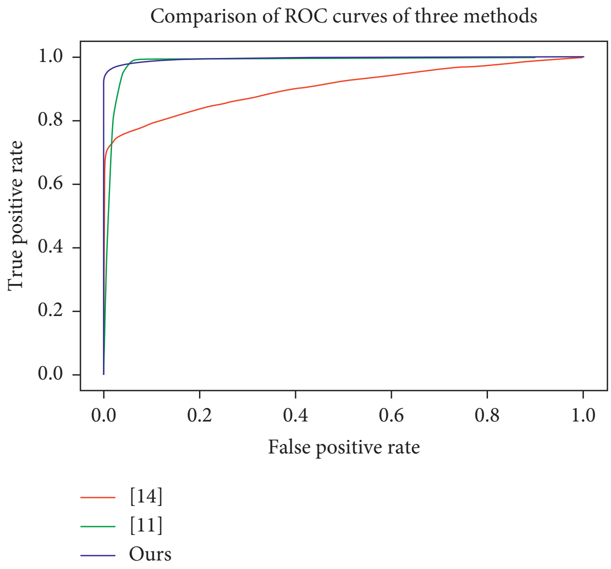


Figure 9: Comparison of ROC curves of three methods.

when extracting HTTP traffic feature, make the focal loss suitable for HTTP traffic feature, and verify that our method is feasible in other mobile communication protocols.

Data Availability

The authors cannot share their data because the data are confidential.

Conflicts of Interest

All authors declare that there are no conflicts of interest.

Authors' Contributions

Wenmin Li and Sanqi Sun made substantial contributions to the drafting of the manuscript. Wenmin Li and Shuo Zhang made contributions to the outlining and editing of the manuscript. Hua Zhang and Yijie Shi made substantial contribution in the revision process. Wenmin Li and Shuo Zhang made substantial contribution in giving final approval of the submitted version and the revised version to be submitted.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) (grant nos. 62072051, 61671082, 61672110, 61976024, 61972048, and 61802025).

References

[1] China's National Bureau of Statistics. Report on the 70th Anniversary of the Founding of the People's Republic of China[EB/OL]. http://www.stats.gov.cn/tjsj/zxfb/201908/t20190813_1690833.html,2019-08-13.

[2] 360 Internet Security Center. China Internet security report for the first half of 2018 [EB/OL]. http://zt.360.cn/1101061855.php?dtid=1101062360&did=491357630,2018-07-30.

[3] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Information Fusion*, vol. 42, pp. 146–157, 2018.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, June 2014.

[5] J. Wu, Y. Yu, C. Huang, and K. Yu, "Deep multiple instance learning for image classification and auto-annotation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015.

[6] A. Rattani, R. Derakhshani, and A. Ross, *Selfie Biometrics: Advances and Challenges*, Springer International Publishing, Berlin, Germany, 2019.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, Cham, Switzerland, October 2015.

[8] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," in *Proceedings of ACM SIGCOMM'07*, pp. 7–16, Kyoto, Japan, August 2007.

[9] Y. Jin, N. Duffield, J. Erman, P. Haffner, and S. Sen, "A modular machine learning system for flow-level traffic classification in large networks," *The ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, p. 4, 2012.

[10] Y. Lim, H. Kim, J. Jeong, C. K. Kim, and T. T. Kwon, "Internet traffic classification demystified: on the sources of the discriminative power," in *Proceedings of the 2010 ACM International Conference on Emerging Networking Experiments and Technologies*, p. 9, Philadelphia, PA, USA, November 2010.

[11] S. Park, M. Kim, and S. Lee, "Anomaly detection for http using convolutional autoencoders," *IEEE Access*, vol. 6, pp. 70884–70901, 2018.

[12] K. M. Ting, "An instance-weighting method to induce cost-sensitive trees," *IEEE Trans. Knowl. Data Eng.*vol. 14, no. 3, pp. 659–665, 2002.

[13] Z. Chen, Q. Yan, H. Han et al., "Machine learning based mobile malware detection using highly imbalanced network traffic," *Information Sciences*, vol. 433-434, no. 3, 2017.

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," arXiv preprint arXiv: 1708.02002, 2017.

[15] V. Tong, "A novel QUIC traffic classifier based on convolutional neural networks," in *Proceedings of 2018 IEEE Global Communications Conference IEEE*, Abu Dhabi, UAE, December 2018.

[16] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: mobile encrypted traffic classification using multimodal deep learning," *Computer Networks*, vol. 165, pp. 106944.1–106944.12, 2019.

[17] L. Mohammad, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[18] G. Bovenzi, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proceedings of IEEE Globecom 2020*, IEEE, Waikoloa, HI, USA, December 2020.

[19] G. Aceto, "Know your big data trade-offs when classifying encrypted mobile traffic with deep learning," in *Proceedings of 2019 Network Traffic Measurement and Analysis Conference (TMA)*, IEEE, Paris, France, June 2019.

[20] M. Zaharia, M. Chowdhury, T. Das et al., "Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation. USENIX Association*, Berkeley, CA, USA, April 2012.

[21] R. Perdisci, W. Lee, and N. Feamster, "Behavioral clustering of http-based malware and signature generation using malicious network traces," *NSDI*, vol. 10, p. 14, 2010.

[22] H. Zhang, H. Guan, H. Yan et al., "Webshell traffic detection with character-level features based on deep learning," *IEEE Access*, vol. 6, pp. 75268–75277, 2018.

[23] The Apache Software Foundation, Apache Hadoop[EB/OL]. http://hadoop.apache.org, 2019.

[24] J. Kreps, N. Narkhede, and J. Rao, "Kafka: a distributed messaging system for log processing," in *Proceedings of the 6th International Workshop on Networking Meets Databases*, Athens, Greece, June, 2011.

[25] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," *CVPR*, vol. 2, p. 5, 2016.

[26] Q. Song, Y. Guo, and M. Shepperd, "A comprehensive investigation of the role of imbalanced learning for software defect prediction," *IEEE Transactions on Software Engineering*, vol. 99, 2018.

[27] A. J. Vickers and E. B. Elkin, “Decision curve analysis: a novel method for evaluating prediction models,” *Medical Decision Making*, vol. 26, no. 6, pp. 565–574, 2006.

[28] D. M. Powers, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation,” *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5536722, 12 pages
https://doi.org/10.1155/2021/5536722

Research Article

Towards a Statistical Model Checking Method for Safety-Critical Cyber-Physical System Verification

Jian Xie[1,2,3] **Wenan Tan,**[1,2,3] **Bingwu Fang**[2,4] **and Zhiqiu Huang**[1,2,3]

[1] *College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China*
[2] *Key Laboratory of Safety-Critical Software, Nanjing University of Aeronautics and Astronautics, Nanjing, China*
[3] *Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China*
[4] *College of Information Engineering, Anhui Finance and Trade Vocational College, Hefei, China*

Correspondence should be addressed to Bingwu Fang; bingwufang@163.com

Received 10 February 2021; Revised 2 April 2021; Accepted 16 April 2021; Published 18 May 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Jian Xie et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Safety-Critical Cyber-Physical System (SCCPS) refers to the system that if the system fails or its key functions fail, it will cause casualties, property damage, environmental damage, and other catastrophic consequences. Therefore, it is vital to verify the safety of safety critical systems. In the community, the SCCPS safety verification mainly relies on the statistical model checking methodology, but for SCCPS with extremely high safety requirements, the statistical model checking method is difficult/infeasible to sample the extremely small probability event since the probability of the system violating the safety is very low (rare property). In response to this problem, we propose a new method of statistical model checking for high-safety SCCPS. Firstly, with the CTMC-approximated SCCPS path probability space model, it leverages the maximum likelihood estimation method to learn the parameters of CTMC. Then, the embedded DTMC can be derived from CTMC, and a cross-entropy optimization model based on DTMC can be constructed. Finally, we propose an algorithm of iteratively learning the optimal importance sampling distribution on the discrete path space and an algorithm to check the statistical model of verifying the rare attribute. Eventually, experimental results show that the method proposed in this paper can effectively verify the rare attributes of SCCPS. Under the same sample size, comparing with the heuristic importance sampling methods, the estimated value of this method can be better distributed around the mean value, and the related standard deviation and relative error are reduced by more than an order of magnitude.

1. Introduction

Safety-Critical Cyber-Physical System (SCCPS) is characterized with high safety and high reliability and are widely used in fields closely related to the national economy and people's livelihoods, such as aerospace, nuclear industry, public transportation, finance, and medical care. Once the execution of such system fails, it will deeply threaten the safety of human's life and property [1–3]. Therefore, it is vital to analyze and verify the safety and reliability of safety-critical systems, and it is of great significance to the design and development of safety-critical systems. Indeed, it has attracted wide attention from researchers and has extensively grown as a prominent research topic in the community [4–7].

Essentially, SCCPS is a kind of complex cyber-physical fusion system [8–10]. For this kind of systems, the continuously changing behavior in their physical layer is intertwined with the discrete changing behavior in their decision control layer. Their state spaces are infinite as well. It increases the difficulty and brings severe challenges to the safety analysis and verification of SCCPS. However, the traditional model checking has the problem of state space explosion, and it is difficult to effectively verify it [11].

With the execution path of the sampling system, Statistical Model Checking (SMC) uses statistical analysis techniques to approximate the probability that the target system meets the sequential logic attributes and can provide arbitrarily small error limits [12–14]. Because SMC does not need to analyze the complex logic inside the target system to verify the timing logic properties of the system, it can effectively avoid the complexity of the system and the explosion of the state space [15, 16]. Therefore, SMC is the most effective solution to verify the timing properties of complex SCCPS [12, 17–19]. However, for SCCPS requiring extremely high safety, the probability of occurrence of the negative events of its safety attributes and the probability of system failures are extremely low. It is infeasible for SMC to sample extremely low probability events. Thus, how to use SMC to verify the extremely secure SCCPS is an urgent problem to be solved [20, 21].

To date, verification of the SMC rare attributes mainly relies on the importance sampling method. For CTMC and DTMC random models, Reijsbergen et al. [22] and Barbot et al. [23] utilized the heuristic methods to obtain an importance sampling distribution to complete the attribute verification of the two models, respectively. Clarke and Zuliani [24] proposed the cross-entropy minimization importance sampling-based SMC method to verify the safety properties of the Stateflow/Simulink model system. Zuliani et al. [17] used the SMC method in his study [24] to verify the secure attribute of the discrete-time SHS. The methods proposed by Clarke and Zuliani assume that the distribution of the system path space is an exponential distribution. By simply increasing the failure rate of the system parameters, several paths that satisfy the rare attributes are extracted at one time to calculate the optimal parameters for the exponential distribution to obtain an importance sampling distribution [25]. J´egourel et al. [26] leveraged the cross-entropy minimum optimization method in the random model of a random guardian command system, which can approximate the path distribution of the system by increasing the number of commands (number of parameters), to obtain an importance sampling distribution in the random model. However, the optimal importance sampling distribution obtained with the aforementioned methods is not from the distribution family of the system path space, but essentially is a heuristic importance sampling method. Thus, the verification results are only rough approximation.

In this paper, we propose a method with the SCCPS path space to construct a cross-entropy optimization model and use an iterative learning method to obtain an optimal importance sampling distribution from the parameterized distribution cluster of the path space. It can ensure that the optimal importance sampling distribution is from the spatial distribution family in the SCCPS path, and the iterative learning method can ensure that the distribution evenly covers the unsafe path distribution area. As evaluated in our experiments, the accuracy and efficiency of the rare attribute verification are significantly improved.

2. Background

2.1. Statistical Model Checking. Statistical Model Checking (SMC) can be simply described as follows: given a system model M and system properties φ described by the bounded linear temporal logic (BLTL) [18], it uses the Monte Carlo sampling, model checking, and statistical analysis techniques to qualitatively/quantitatively verify the following two questions:

(i) The probability that M satisfies the attribute φ: $\Pr(M \vDash \pi)$

(ii) Whether the probability of M satisfying the attribute φ is higher than or equal to the threshold θ: $M \vDash \Pr(\geq \theta)\ (\varphi)$

In SMC, it first simulates the execution of the system model M to extract a random execution path ω. Then, the BLTL model detector is used to determine whether ω satisfies the attribute φ, and a certain number of samples will be generated after multiple simulations. It further leverages the statistical method to perform statistical analysis on the samples to assess the probability of the system model M satisfying the attribute φ, as well as give the confidence interval or the estimated error margin. Let $I(\omega)$ represent the output result of the BLTL model detector. If $\omega \vDash \pi$, $I(\omega) = 1$; otherwise, it is 0. $I(\omega)$ is a Bernoulli random variable, so the behavior of M can be modeled by the Bernoulli distribution with a parameter p:

$$\begin{cases} \Pr(I(\omega) = 1) = p, \\ \Pr(I(\omega) = 0) = 1 - p. \end{cases} \tag{1}$$

The parameter p represents the probability that the model M satisfies the BLTL attribute φ. With the Bernoulli distribution, we note that $p = E[I(\omega)]$, $\mathrm{var}[I(\omega)] = p \times (12212p)$. Since the value of p is unknown, the goal of SMC is to estimate the value of p.

SMC can be divided into two categories: hypothesis testing and parameter estimation. The hypothesis testing is used to determine whether the probability of the system satisfying the temporal logic attribute is greater than or equal to a given threshold, which is a qualitative result, while the parameter estimation is a quantitative result to represent the approximate probability of the system satisfying the temporal logic attribute. SMC qualitative algorithms include the single sampling plan (SSP) algorithm [27], the sequential probability ratio test (SPRT) algorithm [27], and the Bayesian hypothesis test (BHT) algorithm [18]. SMC quantitative algorithms mainly include the approximate probabilistic model checking (APMC) [28] algorithm and the Bayesian interval estimation testing (BIET) algorithm [18]. Kim et al. [29] conducted an empirical evaluation on the performance and applicability of the four algorithms (i.e., SSP, SPRT, BHT, and BIET).

2.2. Safety Requirement Specification. In this paper, we use Bounded Linear Temporal Logic (BTCL) as our specification language. BLTL restricts Linear Temporal Logic (LTL) with time bounds on the temporal operators. Formally, the syntax of BLTL is given as

$$\varphi ::= x \sim v | (\varphi\_1) \vee \varphi\_2 | (\varphi\_1 \wedge \varphi\_2) | \varphi\_1 \varphi\_1 \cup^{t} \varphi\_2, \tag{2}$$

where $\sim \in \{\leq, \geq, =\}$, $x \in SV$ (the finite set of state variables), $v \in \mathbb{R}$, $t \in \mathbb{R}_{\geq 0}$, and $\vee$, $\wedge$, and are the usual Boolean connectives. The formulas $x \sim v$ is called the atomic propositions (AP). The formula $\varphi\{\_1\} \cup^{t} \varphi\{\_2\}$ will return true if and only if $\varphi\{\_2\}$ is true and $\varphi\{\_1\}$ will hold within the time t. The operators $\Diamond_t$ and $\Box_t$ can be defined as follows by using the $\cup t$ operator: $\Diamond t\varphi =$ True $\cup t\varphi$, which required φ to hold true within time t (true). $\Box t\varphi = \neg\Diamond t\neg\varphi$ requires φ to hold true up to time t.

The semantics of BLTL formulas [28, 30, 31] is defined with respect to system traces (or executions). A trace is a sequence $\sigma = (s_0, t_0), (s_1, t_1), \ldots$, where s_i is the state of the system at the represented time t_i. The pair (s_i, t_i) expresses the fact that the system moved to state s_{i+1} after having spent t_i time units in state s_i. If the trace σ satisfies the property φ, we write $\sigma\vDash\varphi$. The trace suffix of σ starting at $k \in \mathbb{N}$ is denoted by σ^k, and σ^0 denotes the full trace σ.

The semantics of BLTL for a trace σ^k is defined as follows:

(i) $\sigma^k\vDash x \sim v$, iff $x \sim v$ holds true in state s_k

(ii) $\sigma^k\vDash\varphi_1\wedge\varphi_2$, iff $\sigma^k\vDash\varphi_1$ and $\sigma^k\vDash\varphi_2$

(iii) $\sigma^k\vDash\varphi_1\vee\varphi_2$, iff $\sigma^k\vDash\varphi_1$ or $\sigma^k\vDash\varphi_2$

(iv) $\sigma^k\vDash\varphi_1$, iff $\sigma^k\vDash\varphi_1$ does not hold ($\sigma^k\nvDash\varphi_1$)

(v) $\sigma^k\vDash\varphi\_1\cup^t\varphi\{\_2\}$, iff $\exists i \in \mathbb{N}$ such that (a) $\sum_{l=0}^{i-1} t_{k+1} < t$ and (b) $\sigma^{k+i}\vDash\varphi_2$, as well as (c) $\forall 0 \le j < i, \sigma^{k+j}\vDash\varphi_1$

The sampling bound: $\#(\varphi)\in\in Q \ge 0$ of a BLTL formula φ is the maximum nested sum of time bounds

(vii) $\#(x \sim v):\ = 0$

(viii) $\#(\varphi_1):\ = \#(\varphi)$

(ix) $\#(\varphi_1\vee\varphi_2):\ = \max(\#(\varphi_1), \#(\varphi_2))$

(x) $\#(\varphi_1\wedge\varphi_2):\ = \max(\#(\varphi_1), \#(\varphi_2))$

(xi) $\#(\varphi_1\cup^t\varphi_2):\ = t + \max(\#(\varphi_1), \#(\varphi_2))$

Lemma 1 (Bounded sampling). *The problem "σ⊨φ" is well-defined and can be checked for BLTL formulas φ and traces σ based on only a finite prefix of σ of bounded duration.*

Proof. According to Lemma 1, the decision "$\sigma\vDash\varphi$" is uniquely determined (and well-defined) by considering only a prefix of σ of duration $\#(\varphi)\in\in Q \ge\ 0$. By divergence of time, σ reaches or exceeds this duration $\#(\varphi)$ in some finite number of steps n. Let σ^0 denote a finite prefix of σ of length n, such that $\sum_{0\le l<n} tl \ge \#(\varphi)$. Again by Lemma 3, the semantics of $\sigma^0\vDash\varphi$ is well-defined because any extension σ" of σ' satisfies σ"⊨φ if and only if σ'⊨φ. Consequently, the semantics of σ'⊨φ coincides with the semantics of $\sigma \vDash\varphi$. On the finite trace σ^0, it is easy to see that BLTL is decidable by evaluating the atomic formulas $x \sim v$ at each state s_i of the system simulation. □

Lemma 2 (BLTL on bounded simulation traces). *Let φ be a BLTL formula, $k \in N$. Then, for any two infinite traces, $\sigma = (s_0, t_0), (s_1, t_1), \ldots$ and $\overline{\sigma} = (\overline{s}_0, \overline{t}_0), (\overline{s}_1, \overline{t}_1), \ldots$ with $s_{k+I} = \overline{s}_{k+I}$ and $t_{\underline{k+I}} = \overline{t}_{k+I}\ \forall I \in N$ with $\sum_{t_{k+I}} \le \#(\phi)$ [17]. We have that $\sigma_k\vDash\varphi$ if $\overline{\sigma_k}\vDash\varphi$.*

Proof. IH is short for induction hypothesis.

(1) If φ is of the form $x \sim v$, $\sigma_k\vDash\varphi$ if $\overline{\sigma_k}\vDash\varphi$ since $s_{k+I} = \overline{s}_{k+I}$ and $t_{k+I} = \overline{t}_{k+I}$ by using [17] for $i = 0$.

(2) If φ is of the form $\varphi_1\vee\varphi_2$,

$$\sigma_k\vDash\varphi_1\vee\varphi_2 \begin{cases} \text{iff } \sigma_k\vDash\varphi_1 \text{ or } \sigma_k\vDash\varphi_2, \\ \text{iff } \overline{\sigma}_k\vDash\varphi_1 \text{ or } \overline{\sigma}_k\vDash\varphi_2, \\ \text{iff } \overline{\sigma}_k\vDash\varphi_1\vee\varphi_2, \end{cases} \tag{3}$$

by induction hypothesis as $\#(\varphi_1\vee\varphi_2) \ge \#(\varphi_1)$ and $\#(\varphi_1\vee\varphi_2) \ge \#(\varphi_2)$. The proof is similar to φ_1 and $\varphi_1 \cap \varphi_2$.

(3) If φ is of the form $\varphi_1\cup^t\varphi_2$, $\sigma_k\vDash\varphi_1\cup^t\varphi_2$ if the following three conditions are satisfied:

(a'). $\sum_{0\le l<i} t_{\tilde{k+l}} \le t$ because $\#(\varphi_1\cup^t\varphi_2) \ge t$ such that the durations of trace σ and $\overline{\sigma}$ are $t_{k+l} = t_{\tilde{k+l}}$ for each index l with $0 \le l < i$ by the assumption [17].

(b'). $\overline{\sigma}_{k+i}\vDash\varphi_2$ by induction hypothesis as follows: we know that the traces σ and $\overline{\sigma}$ match at k for duration $\#(\varphi_1\cup^t\varphi_2)$ and need to show that the semantics of $\varphi_1\cup^t\varphi_2$ matches at k. By IH, we know that φ_2 has the same semantics at $k + i$ (that is, $k + i\vDash\varphi_2$ if $k + i\vDash\varphi_2$) provided that we can show that the traces σ and $\overline{\sigma}$ match at $k + i$ for duration $\#(\varphi_2)$. For this case, it considers any $I \in N$ with $\sum_{0\le l<I} t_{k+i+l} \le \#(\varphi_2)$. Then, $\#(\varphi_2) \ge \sum_{0\le l<I} t_{k+i+l} = \sum_{0\le l<}\ It_{k+l} - \sum_{0\le l<I} t_{k+l}\ \ge \sum_{0\le l<i+l} t_{k+l} - t$. Thus, $\sum_{0\le l<i+I} t_{k+l} \le t + \#(\varphi_2) \le\ \ t + \max(\#(\varphi_1), \#(\varphi_2)) = \#(\varphi_1\cup^t\varphi_2)$. As $I \in N$ was arbitrary, we conclude from this with assumption [17] that, indeed $s_I = s_{\tilde{I}}$ and $t_I = t_{\tilde{I}}$ for all $I \in N$ with $\sum_{0\le l<I} t_{k+i+l} \le \#(\varphi_2)$. Thus, the IH for φ_2 yields the equivalence of $\sigma_{k+i}\vDash\varphi_2$ and $\overline{\sigma}_{k+i}\vDash\varphi_2$ when using the equivalence of (a) and (a').

(c'). For each $0 \le j < i$, $\sigma_{k+i}\vDash\varphi_1$. The proof of equivalence to (c) is similar to that for (b') using $j < i$. The existence of an $i \in N$ for which these conditions (a'), (b'), and (c') hold is equivalent to $k\vDash\varphi_1\cup^t\varphi_2$. □

2.3. Safety Critical System Model. Safety-Critical Systems (SCCPS) [32] are defined as a tuple, SCCPS = (L, X, E, Inv, D, G, R), where

(i) L is a finite set of discrete states (control mode);

(ii) $X \subseteq \mathbb{R}^n$ is a finite set of continuous variables;

(iii) $E \subset L \times L$ is a collection of discrete changes;

(iv) Inv: $L \longrightarrow 2^X$ represents the mapping from the discrete state set L to the continuous state space. For $\forall l \in L$, Inv (l) is the invariant-position set of l;

(v) D: $L \longrightarrow (X \longrightarrow X)$ is a mapping of a vector domain, which assigns a set of Stochastic Differential Equations (SDE) to each control mode $l \in L$ to describe the continuous random dynamic behavior with respect to the different control modes l, $d_x(t) = f(l, x(t))d_t + g(l, x(t))d_{B_t}$. B_t is a standard Wiener process defined in the real number field. It assumes that $\forall l \in L$, $f(l,\cdot)$, and $g(l,\cdot)$ are bounded and Lipschitz continuous;

(vi) G: $E \longrightarrow 2^X$ is to assign a guardian condition to each discrete transition, satisfying the following conditions:

** $\forall e = (l,l') \in E, G(e)$ denotes a measurable subset of ∂ Inv (l)

** $\forall l \in L, \{G(e)$: $e = (l,l') \in E, l' \in L\}$ is a disjoint subset of ∂ Inv (l)

(vii) R: $E \times X \longrightarrow \mathscr{P}(X)$ is a reset mapping. $\mathscr{P}(X)$ represents a set of probability measures defined on X, and continuous variables are reset according to the probability distribution.

According to the definition, the SCCPS hybrid state space is $L \times X$, and $(l,x) \in L \times X$ represents the hybrid state. The continuous dynamics of SCCPS evolves according to the SDE in the current control mode. However, the discrete dynamics refers to migrating one control mode to another control mode with the guardian condition on the discrete transition, when the continuous variable cannot reach the boundary of the invariant.

Let $x_l(t)$ be the SDE solution of the initial state $x_l(0)$; $\tau(l) = \inf\{t \in \mathbb{R}_{>0}, x_l(t) \notin \text{Inv}(l)\}$ means that, in the control mode l, the first time that the evolution of a continuous variable violates the invariant, that is, the first time of exiting the control mode l.

SCCPS execution semantics: a random execution of SCCPS is defined as a random process $(l(t), x(t)) \in L \times X$ in the SCCPS state space. If there is a stop-time sequence $T_0 = 0 < T_1 < T_2 < \cdots$ that makes $\forall k \in \mathbb{N}$, where

(i) $(l_0, x_0) \in L \times X$ indicates the initial state of SCCPS.

(ii) $t \in (T_k, T_{k+1}), l(t) = l(T_k)$ is a const, and $x(t)$ is a continuous solution of the SDE $d_x(t) = f(l(T_k), x(t))d_t + g(l(T_k), x(t))d_{B_t}$;

- $T_{k+1} = T_k + \tau(l(T_k))$;
- the probability distribution of $x(T_{k+1})$ is determined by the reset map $R(e_k, x(T_{k+1}^-))$, where $e_k = (l(T_k), l(T_{k+1})) \in E$ and $x(T_{k+1}^-) = \lim_{t\longrightarrow T_{k+1}} x(t)$.

SCCPS path: a SCCPS execution path is defined as an infinite sequence $\sigma = ((l_0,x_0),t_0), ((l_1,x_1),t_1), \ldots$ from the initial state (l_0,x_0), where $(l_i,x_i) \in L \times X$ represents the SCCPS state. $t_i \in R_{\geq 0}$ means the time that transitions the state (l_i,x_i) to the next state (l_{i+1},x_{i+1}).

3. Our Approach

In this section, we present our proposed method with the SCCPS path space to construct a cross-entropy optimization model and use an iterative learning method to obtain an optimal importance sampling distribution from the parameterized distribution cluster of the path space.

3.1. SCCPS Path Space Model

3.1.1. Model Representation. To avoid the complexity of the dynamic evolution of SCCPS, SMC does not pay attention to the structure of SCCPS, but focus on sampling the execution path of SCCPS. The behavior of SCCPS evolving over time can be characterized by the path of the system. According to the execution semantics of SCCPS, the execution path generation process of SCCPS can be described as follows: in the current control mode l_i, the continuous variable x_i evolves according to the SDE. When the evolution of x_i satisfies the guardian condition $(x_i \in G(l_i, l_{i+1}))$, it migrates to the next control mode l_{i+1} and the initial value of x_{i+1} is determined by the random reset kernel R. The residence time of l_i is $t_i = \inf\{t \in R_{>0}, x_i(t) \notin \text{Inv}(l_i)\}$. t_i is a random variable, and its value depends on the SDE of l_i and the initial values $x_i(0)$ and Inv (l_i). According to the generation process of the SCCPS execution path, the next state of SCCPS depends on the current state and the related residence time of the current state. Therefore, the execution path of the SCCPS can be regarded as that it is generated in the continuous-time Markov process in the hybrid state space. As the residence time of l_i is longer, the probability of migration from l_i is higher. It can further presume that the residence time of l_i obeys the exponential distribution, and the continuous-time Markov process then becomes CTMC.

Let G_l denote the guard condition set of all edges starting from l:

$$G_l = \{G(e)\colon e = (l,l') \in E, l' \in \text{Loc}\}, \tag{4}$$

where $G(e) \in \partial$ Inv (l) and $G(e_i) \cap G(e_j) = \varnothing$, $i \neq j$. In l, the time for the continuous variable evolving to satisfying the conditions of each guard is $\tau_1, \tau_2, \ldots, \tau_{|G_l|}$. Then, the residence time in l is $t_l = \min\{\tau_1, \tau_2, \ldots, \tau_{|G_l|}\}$. Supposing $\tau_1, \tau_2, \ldots, \tau_{|G_l|}$, respectively, obey the exponential distribution of parameters $\{\lambda_{l,l'}, l' \in L, (l,l') \in E\}$, then the residence time t_l in l obeys the exponential distribution of parameters $\sum_{l' \in \text{Loc}, (l,l') \in E} \lambda_{l,l'}$. With this assumption, the execution path of SCCPS can be generated by the CTMC random process.

Definition 1. SCCPS path generation model: the path generation model on the SCCPS state space is defined as CTMC = (S, s_0, λ), where

(i) $S = L$ represents the discrete state set of SCCPS

- $s_0 \in L$ denotes the initial state of SCCPS
- Migration rate function λ: $S \times S \longrightarrow R_{\geq 0}$, and all migration rate function values form the migration rate matrix λ

It can be seen from this definition that when the CTMC structure is known, its behavior is controlled by the migration rate matrix λ, whose value comes from SCCPS. The value of λ is estimated with the maximum likelihood method according to simulating the execution of SCCPS to obtain the time samples of the state transition.

3.1.2. Algorithm of Learning Model Parameters. The rarity of the path does not necessarily imply that the conversion rate between two adjacent discrete states is low, and the rarity of the safety attributes in the path space does not necessarily imply that the optimal parameters in the parameter space are rare. Based on this observation, this section introduces our approach of leveraging the maximum likelihood estimation method to estimate the migration rate of two adjacent discrete states of SCCPS and obtain the migration rate matrix λ. With the simulation operation of each discrete state of SCCPS, the discrete state is sampled to migrate to the next discrete state time; we then use the maximum likelihood estimation to obtain an estimate of λ.

For the state $s_i \in S$, we simulate executing the SDE in the running state s_i to obtain the migration time $t_k (k = 1, \ldots, N)$ samples of the adjacent state s_j. Assuming that the migration time between s_i and s_j obeys the exponential distribution of the parameter λ_{ij}, then the likelihood function of λ_{ij} can be obtained:

$$L\left(\lambda_{ij}\right) = \prod_{k=1}^{N} \lambda_{ij} e^{-\lambda_{ij} t_k}, \tag{5}$$

and its log likelihood function is as follows:

$$\ln L\left(\lambda_{ij}\right) = \sum_{k=1}^{N} \ln \lambda_{ij} - \lambda_{ij} \sum_{k=1}^{N} t_k. \tag{6}$$

We further take the derivative of λ_{ij} with the log-likelihood function and make it equal to 0, and its estimated value can be resolved, $\widehat{\lambda}_{ij} = (1/N)\sum_{k=1}^{N} t_k$. With $E(\widehat{\lambda}_{ij}) = (1/N)\sum_{k=1}^{N} E(\widehat{\lambda}_{ij}) = (1/\lambda_{ij})$, it can be seen that the estimated value is an unbiased estimate of λ_{ij}. The estimated variance is

$$\operatorname{Var}\left(\widehat{\lambda}_{ij}\right) = \operatorname{Var}\left(\frac{1}{N}\sum_{k=1}^{N} t_k\right) = \frac{1}{N^2}\sum_{j=1}^{N} \operatorname{Var}\left(t_k\right) = \frac{1}{N\lambda_{ij}^2}, \tag{7}$$

but the estimated variance is biased, and the variance will be decreased as the samples increase.

In most cases, it is difficult to obtain a clear expression for the random execution of SCCPS. However, what the safety concerned is the accessibility analysis of discrete states. The discrete state set S and its transitions can capture all necessary information. Therefore, we derive the DTMC from the SCCPS path generation model to represent the path space of SCCPS. The value of DTMC's migration probability matrix $P\colon S \times S \longrightarrow [0,1]$ can be obtained from the migration rate matrix λ of the SCCPS path generation model. For two states s_i and $s_j \in S$,

$$P\left(s_i, s_j\right) = \begin{cases} \dfrac{\lambda_{ij}}{\lambda_i}, & s_i \neq s_j, \\[2ex] 1, & s_i = s_j, \end{cases} \tag{8}$$

where $\lambda_i = \sum_{s_j \in S} \lambda_{ij}$.

3.2. Method of Sampling Rare Attributes. In the path space of the high-safety SCCPS, it is difficult to obtain samples satisfying the rare attributes, which makes the SMC infeasible. To address this challenge, we propose a method for sampling the rare attributes. It uses the cross-entropy method to learn an optimal-importance sample distribution from the path space of the SCCPS. With this sample distribution, it is easy to obtain the samples that satisfy the rare attributes. Thus, the convergence of the SMC can be accelerated. The importance sampling distribution is corrected by the likelihood ratio weighting to ensure that the SMC verification result is unbiased.

3.2.1. Zero-Variance Importance Sampling Distribution. The basic idea of the importance sampling method [33, 34] is to change the probability density distribution of random variables, so as to obtain the samples of extremely small probability events with a higher probability. We now present the SMC method based on the importance sampling. Let $f(\omega)$ be the true distribution of path ω, and let $g(\omega)$ be the importance sampling distribution, and $g(\omega)$ can obtain the samples of the extremely small probability events with a higher probability when $g(\omega) \neq 0$ and $f(\omega) \neq 0$. In the case of verifying the extremely small probability events, it is difficult to sample from $f(\omega)$ to meet the requirements, but the importance sampling method is to sample from $g(\omega)$. The probability $p = E_f[I(\omega)]$ satisfying the system attribute can be described as

$$\begin{aligned} p = E_f[I(\omega)] &= \int I(\omega) f(\omega) \mathrm{d}_\omega = \int I(\omega) \frac{f(\omega)}{g(\omega)} g(\omega) \mathrm{d}_\omega \\ &= \int I(\omega) W(\omega) g(\omega) \mathrm{d}_\omega = E_g[I(\omega) W(\omega)], \end{aligned} \tag{9}$$

where $W(\omega) = (f(\omega)/g(\omega))$ is the likelihood ratio, and $g(\omega)$ is for the importance sampling. We leverage the likelihood ratio to correct the weighting to ensure that the estimated value of p is unbiased. We then randomly sample N independent execution paths $\omega_i, i \in \{1, \ldots, N\}$ from the importance distribution $g(\omega)$ and obtain the unbiased estimate:

$$\widehat{p} = \frac{1}{N} \sum_{i=1}^{N} I(\omega_i) W(\omega_i), \tag{10}$$

and estimated variance

$$\operatorname{Var}_g[\widehat{p}] = \frac{1}{N}\left(E_g\left[I^2(\omega) W^2(\omega)\right] - p^2\right), \tag{11}$$

for p, respectively.

The efficiency and accuracy of importance sampling rely on the selection of the distribution $g(\omega)$. If the selection is inadequate, the importance sampling method is difficult to effectively achieve the acceleration effect and may play a decelerating effect. The key problem of importance sampling is to find a density function for the optimal sampling probability to minimize the estimated variance. With formula (10) returning 0, it can obtain the following formula:

$$g^*(\omega) = \frac{I(\omega) f(\omega)}{p}, \tag{12}$$

where $g^*(\omega)$ is a zero-variance importance sampling distribution, which means that extracting only one sample from the zero-variance importance sampling distribution can be used to calculate its estimated value, that is, any sample is an unbiased estimate of its mean. However, the zero-variance importance sampling distribution depends on the true value p, and the value of p is unknown. Therefore, it is impossible to sample from $g^*(\omega)$. This paper proposes to use the cross-entropy method to find an approximate optimal importance sampling distribution closest to $g^*(\omega)$ from the parameterized distribution family of the sample path space, so as to reduce the SMC variance and accelerate the convergence of the SMC algorithm.

3.2.2. Cross-Entropy Optimization Model. This section is to obtain the optimal importance sampling distribution by minimizing the cross entropy between the two probability distributions. According to the definition of cross entropy [35], this section provides the definition of cross entropy for the SCCPS path space.

Definition 2. Cross entropy for the SCCPS path space: the cross entropy between two probability measures $f(\omega)$ and $f'(\omega)$ for the SCCPS path space Ω is as follows:

$$\mathrm{CE}\left(f(\omega), f'(\omega)\right) = \int_{\Omega} f(\omega) \ln \frac{f(\omega)}{f'(\omega)} \mathrm{d}_{\omega}. \tag{13}$$

The cross entropy is used to assess the similarity of two probability distributions. The value of cross entropy is smaller, and $f(\omega)$ and $f'(\omega)$ are more similar, i.e., $\mathrm{CE}(f(\omega), f'(\omega)) = 0$ if and only if $f(\omega) = f'(\omega)$.

According to Definition 2, the construction of the cross-entropy optimization model on the SCCPS path space is given below. Assume that the original distribution $f(\omega)$ of the SCCPS path ω comes from the parameterized distribution family $\{f(\omega, \theta)\}$, The cross-entropy optimization method is used to select a distribution $f(\omega, \lambda^*)$, $\lambda^* \in \theta$ in the parameterized distribution family, $\lambda^* \in \theta$ and the optimal distribution $g^*(\omega)$ have the smallest cross-entropy. This optimization problem can be described for

$$\begin{aligned} \min_{\lambda} \mathrm{CE}\left(g^*(\omega), f(\omega, \lambda)\right) &= \min_{\lambda} \int_{\Omega} g^*(\omega) \ln \frac{g^*(\omega)}{f(\omega, \lambda)} \mathrm{d}_{\omega} \\ &= \min_{\lambda} \int_{\Omega} g^*(\omega) \ln g^*(\omega) \mathrm{d}_{\omega} \\ &\quad - \int_{\Omega} g^*(\omega) \ln f(\omega, \lambda) \mathrm{d}_{\omega}. \end{aligned} \tag{14}$$

The first term of formula (13) has nothing to do with λ and minimizing cross entropy is equivalent to maximizing the second term. Let $D(\lambda) = \int_{\Omega} g^*(\omega) \mathrm{Inf}(\omega, \lambda) \mathrm{d}_{\omega}$; the minimization problem of formula (13) is equivalent to the maximization problem of formula (14):

$$\begin{aligned} \max_{\lambda} \int_{\Omega} g^*(\omega) \ln f(\omega, \lambda) \mathrm{d}_{\omega} &= \max_{\lambda} \int_{\Omega} I(\omega) f(\omega) \ln f(\omega, \lambda) \mathrm{d}_{\omega} \\ &= \max_{\lambda} E[I(\omega) \ln f(\omega, \lambda)]. \end{aligned} \tag{15}$$

Solving the optimization problem of formula (14) requires sampling from the true distribution $f(\omega)$. However, in the case of rare attribute verification, it is difficult to sample from $f(\omega)$ to the path sample that satisfies the rare attribute. By using importance again, the sampling method samples from the distribution $f(\omega, \mu)$ and the selection of parameter μ should be able to increase the probability of the path that meets the rare attribute. Therefore, the optimization problem of formula (14) can be re-formed as

$$\begin{aligned} &\max_{\lambda} \int_{\Omega} I(\omega) \frac{f(\omega)}{f(\omega, \mu)} f(\omega, \mu) \ln f(\omega, \lambda) \mathrm{d}_{\omega} \\ &= \max_{\lambda} \int_{\Omega} I(\omega) W(\omega, \mu) f(\omega, \mu) \ln f(\omega, \lambda) \mathrm{d}_{\omega}. \\ &= \max_{\lambda} E_{\mu}[I(\omega) W(\omega, \mu) \ln f(\omega, \lambda)]. \end{aligned} \tag{16}$$

Among them, the likelihood ratio function $W(\omega, \mu) = (f(\omega)/f(\omega, \mu))$. In formula (16), the optimal solution of its optimization problem λ^* can be estimated by the path sample, and the sample mean is replaced by the expectation Get the estimated value of λ^*

$$\widehat{\lambda^*} = \mathrm{argmax}_{\lambda} \frac{1}{N} \sum_{i=1}^{N} I(\omega_i) W(\omega_i, \mu) \ln f(\omega_i, \lambda), \tag{17}$$

where $\omega_1, \omega_2, \ldots, \omega_N$ is a sample from the distribution $f(\omega, \mu)$.

3.3. Algorithm of Verifying the Cross-Entropy Safety. In Section 3.1, we provide a DTMC-based method to approximate the SCCPS path space. SMC mainly considers the system execution path $\omega = s_0, s_1, \ldots, s_k \, (k > 0)$ within a bounded time T, where k is a random variable to represent the number of state transitions, and its value varies with ω. Let $\langle l, m \rangle$ denote two adjacent and ordered state pairs in ω, $S(\omega)$ represent the set of ordered state pairs in ω, $n_{lm}^{(\omega)}$ represent the number of transitions from state l to state m in ω, and $n_l^{(\omega)}$ represent the number of occurrences of the state l in ω; then, the probability measure function of path ω under system parameter p can be formulated as

$$f(\omega, p) = \iota_{\mathrm{init}}(s_0) \prod_{\langle l,m \rangle \in S[\omega]} \left(p_{lm}\right)^{n_{lm}^{(\omega)}}. \tag{18}$$

Substituting $f(\omega_i, \lambda)$ of formulas (16) with (17), we obtain

$$\max_p \quad \frac{1}{N}\sum_{i=1}^{N} I\left(\omega_i\right) W\left(\omega_i, \mu\right)\left(\operatorname{lnt}_{\text{init}}\left(s_0\right) + \sum_{\langle l,m\rangle\in S\left(\omega_i\right)} n_{lm}^{\left(\omega_i\right)} \ln p_{lm}\right) \text{s.t.} \sum_{m\in S} p_{lm} = 1, \tag{19}$$

and formula (18) can be transformed by the Lagrangian multiplier method into the following optimization problem:

$$\max_p \sum_{i=1}^{N} I\left(\omega_i\right) W\left(\omega_i, \mu\right)\left(\ln \iota_{\text{init}}\left(s_0\right) + \sum_{\langle l,m\rangle\in S\left(\omega_i\right)} n_{lm}^{\left(\omega_i\right)} \ln p_{lm}\right) + \nu_i\left(\sum_{m\in S} p_{lm} - 1\right), \tag{20}$$

where ν_i is the Lagrangian multiplier. Taking the derivative of formula (19) to p_{lm} and making it equal to 0, the solution can be

$$p_{lm} = \frac{\sum_{i=1}^{N} I\left(\omega_i\right) W\left(\omega_i, \mu\right) n_{lm}^{\left(\omega_i\right)}}{\sum_{i=1}^{N} I\left(\omega_i\right) W\left(\omega_i, \mu\right) n_{l}^{\left(\omega_i\right)}}, \tag{21}$$

where ω_i $(1 \leq i \leq N)$ is the sample path from the distribution $f(\omega, \mu)$, and $f(\omega_i)$ represents the true probability distribution of the SCCPS path.

With formula (20), it indicates that the estimated value of the optimal solution relies on the initial distribution $f(\omega, \mu)$. However, the distribution of $f(\omega, \mu)$ is generally far from the optimal distribution. Therefore, in order to reduce the influence of the initial distribution $f(\omega, \mu)$ on the optimal importance sampling distribution, this paper proposes the iterative solution in the path space. Through the iteration, the algorithm can explore a wider path space, so as to obtain a better approximate optimal solution.

Let the initial distribution parameter be $u = p^{(0)}$, and an iterative formula can be obtained from formula (20):

$$p_{lm}^{(j+1)} = \frac{\sum_{i=1}^{N} I\left(\omega_i^{(j)}\right) W\left(\omega_i^{(j)}, p^{(j)}\right) n_{lm}^{\left(\omega_i^{(j)}\right)}}{\sum_{i=1}^{N} I\left(\omega_i^{(j)}\right) W\left(\omega_i^{(j)}, p^{(j)}\right) n_{l}^{\left(\omega_i^{(j)}\right)}}, \tag{22}$$

where N is the number of samples per iteration, $W(\omega_i^{(j)}, p^{(j)}) = (f(\omega_i^{(j)})/f(\omega_i^{(j)}, p^{(j)}))$ represents the likelihood ratio of the nth iteration, and $\omega_i^{(j)}$ is the ith sample path sampled from the distribution $f(\omega_i^{(j)}, p^{(j)})$.

Usually, only a few state transitions can be seen in each simulated execution. During each iteration, some parameters do not work in the path that satisfies the extremely small probability event. Formula (21) will set these parameter values to zero so that these parameters will not work in all subsequent iterations. As a result, the iterative algorithm converges too prematurely to detect a wider parameter space. To avoid this situation, this paper adopts a smoothing strategy to temporarily reduce the importance of inoperative parameters in the iteration instead of simply setting them to zero. The smoothing strategy is to weight current iteration value and the parameters of the previous iteration:

$$p_{lm}^{(j+1)} = \alpha p_{lm}^{(j)} + (1-\alpha)\frac{\sum_{i=1}^{N} I\left(\omega_i^{(j)}\right) W\left(\omega_i^{(j)}, p^{(j)}\right) n_{lm}^{\left(\omega_i^{(j)}\right)}}{\sum_{i=1}^{N} I\left(\omega_i^{(j)}\right) W\left(\omega_i^{(j)}, p^{(j)}\right) n_{l}^{\left(\omega_i^{(j)}\right)}}, \quad \alpha \in (0,1). \tag{23}$$

The smoothing strategy can retain important but not yet effective parameters. Iterative formula (21) and smoothing formula (22) can jointly ensure that approximately uniform sampling is obtained from the path set of events satisfying the minimal probability.

The selected initial distribution $f(\cdot; p^{(0)})$ should be able to produce some paths that satisfy the event with minimal probability in the first iteration, that is, the selected parameter $p^{(0)}$ should be able to increase the probability of occurrence of the extremely small probability events. Therefore, in this paper, we set the initial parameter $p^{(0)}$ to a uniform distribution, and the uniform distribution can quickly obtain the sample path that satisfies the extremely small probability event. The condition for stopping the iteration can be that the coefficient of variance or the distance between two iteration parameter vectors are not higher than a certain constant or the maximum number of iterations. For example, given any small positive number $\epsilon > 0$, if $\|p^{(j)} - p(j-1)\| < \epsilon$ is satisfied, the iteration will be stopped. To facilitate the comparison, we limit the maximum number of iterations in the experiment. To sum up, Algorithm 1 presents the description of the importance sampling distribution learning algorithm, which iteratively solves the approximate optimal importance sampling distribution in the SCCPS path space of the attributes for being verified.

Regardless of sample acquisition time and BLTL model checking time, the time complexity of Algorithm 1 is $O(j_{\max}|p|N)$. Since the optimized objective function is convex, there is a unique optimal solution. If Algorithm 1 can converge, it must converge to the vicinity of the unique optimal solution [36]. Since the number of samples in each iteration is limited, the convergence is probabilistic but not necessarily monotonic. By simply limiting the maximum number of iterations $j_{\max}$, the algorithm can be guaranteed to be terminated with 100% probability. For the proof of convergence of cross-entropy optimization, please refer to [37]; thus, a formal proof of convergence is not provided in this paper. In experiments, we observe that the parameters are convergent. Once the parameters converge, the last set of simulated samples is used to estimate the probability $\hat{p}$ that SCCPS satisfies the safety attribute with the optimal importance sampling distribution. Algorithm 2 describes the verification process of the safety verification algorithm.

4. Experiment and Analysis

To evaluate the effectiveness and performance of the Cross-Entropy Safety Verification Algorithm (CESVA) method proposed in this paper, we apply CESVA to a fault-tolerant controller for an aircraft elevator system (FTC4AE), that is, a Stateflow/Simulink hybrid system modeling case from MATLAB. It introduces the randomness in terms of the fault injection and simulates with MATLAB to obtain the system execution path. Path checking is realized by the BLTL model detector of Plasma-Lab [38]. In the experiment, the rare attributes of FTC4AE is verified with the CESVA method, which is further compared with the Heuristic Importance Sampling (HIS) method [17].

4.1. Validity Measurement of Experimental Results. In the case of nonrare attribute verification, the confidence interval is used to assess the accuracy of various methods, while in the case of rare attribute verification, the relative error of sampling is used to assess the accuracy of the estimation:

$$\mathrm{RE}(\hat{p}) = \frac{\sqrt{\mathrm{Var}[\hat{p}]}}{E[\hat{p}]} \approx \sqrt{\frac{1}{Np}}, \tag{24}$$

where $E[\hat{p}]$ is replaced by the current estimated value $\hat{p}$, $\mathrm{Var}[\hat{p}] = (1/N-1)\sum_{i=1}^{N}(I(\sigma_i)W(\sigma_i,\mu,\lambda^*)-\hat{p})^2$.

Skewness is a measure of assessing the skewing direction and degree of data distribution and is the characteristic number that characterizes the degree of asymmetry of the probability distribution density curve with respect to the average. Skewness is defined as the third-order standardized moment of the sample, and the skewness of the normal distribution is 0, and its estimator is evenly distributed around the mean:

$$\mathrm{skew}(\hat{p}) = \frac{N}{(N-1)(N-2)} \frac{\sum_{i=1}^{N}\left(\hat{p}-(1/N)\sum_{j=1}^{N}\hat{p}_j\right)^3}{(\mathrm{Var}[\hat{p}])^{(3/2)}}. \tag{25}$$

The negative skewness means that the distribution is left-tailed. At this time, the data on the left of the mean are less than the data on the right. Intuitively, the tail on the left is longer than the tail on the right. In contrast, the positive skewness means that the distribution is right-tailed. The data on the right of the mean is less than the left. Intuitively, the tail on the right is longer than the tail on the left.

4.2. Experiment and Analysis on a Fault-Tolerant Controller for the Aircraft Elevator System. The fault-tolerant controller for an aircraft elevator system is a part of a large Simulink model of HL-20 rescuers developed by the National Aeronautics and Space Administration [39]. The two horizontal tails on the two side of the aircraft's fuselage are controlled by two elevators, respectively. Each elevator has two independent hydraulic actuators. In the normal operation process, each elevator is positioned by its corresponding external actuator, and its internal actuator can be used when the external actuator does not work. The two external actuators are driven by two independent hydraulic circuits, and the two internal actuators are both connected to the third hydraulic circuit. The system should ensure that only one set of actuators (i.e., external or internal) locates the elevator at any given time. If the external actuator or its corresponding hydraulic circuit fails, the system will activate the internal actuator. If the fault still exists, the external actuator will be shut down and eventually isolated. The fault in the hydraulic circuit may be temporary, and if the fault is cleared, the hydraulic circuit can always be restored to the online state. The control logic of the system is implemented in the form of a state flow diagram, while the hydraulic actuators and elevators are modeled by using Simulink.

According to modifying the Stateflow/Simulink model, we add random faults into three hydraulic circuits. Setting the fault model with an out-of-bounds' reading of circuit pressure, we model the fault injection as three independent Poisson processes. When the hydraulic circuit fails, the circuit will stay in the fault state for one second. Then, the pressure reading will restore to its normal value, and the fault state will be terminated. In our experiments, the being estimated safety attribute is the probability that, within 25 seconds, the horizontal tails will not respond to the control inputs in the duration of 1 second.

We estimated the probability of the BLTL formula φ:

$$\varphi = F_{25}G_1\left((H_1\mathrm{fail} \vee H_3\mathrm{fail}) \wedge H_2\mathrm{fail}\right), \tag{26}$$

where H_1 and H_3 represent the hydraulic circuit that drives the external actuator, while H_2 represents the hydraulic circuit that drives the internal actuator.

In the experiment, the failure rate of the three hydraulic circuits is set to 0.001, and the failure repair rate is 1. With the two parameters, the parameter v in Algorithm 1 can be calculated. It still is difficult to obtain samples that satisfy the attribute φ with the previous parameters. Therefore, to ensure that the obtained samples can satisfy the attribute φ, the initial failure rate is set as 0.1 and the fault repair rate is set as 1. According to these two parameters, the initial parameter of iteration $p^{(0)}$ in Algorithm 1 can be calculated. In order to assess the performance of verifying the rare attributes with the CESVA method, 20 iterations of Algorithm 1 are performed. In each iteration, the number of samples is $N = 104$, the smoothing factor $\alpha = 0.2$, and the total number of required samples is 2.0×10^5.

Figure 1 shows the change trend of the failure rate parameters during the 20 iterations of the CESVA method. At the beginning of the iteration, the parameters converge rapidly. When the parameters are close to their optimal values, the convergences of their values slow down with random fluctuations. From the 16th iteration, the failure rate parameters start to converge to the stable values. From the

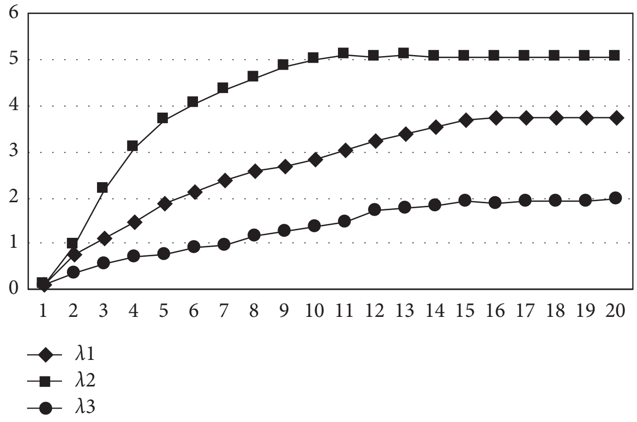


Figure 1: Convergence of parameters during 20 iterations.

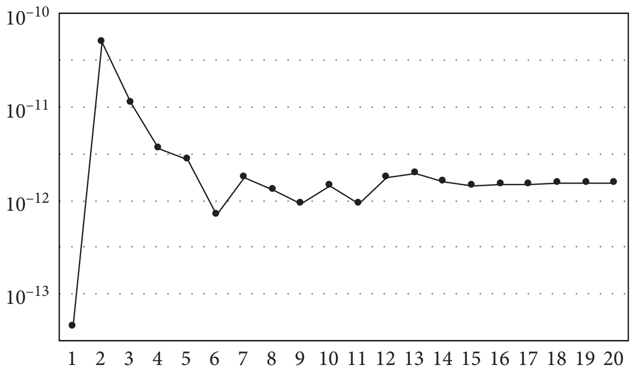


Figure 2: Distribution of estimated values of CESVA during 20 iterations.

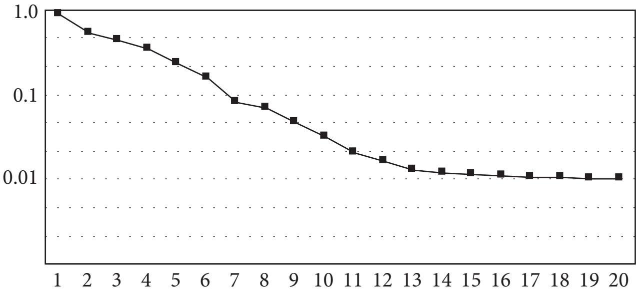


Figure 3: Distribution of relative error of CESVA during 20 iterations.

perspective of the parameter convergence trend, it seems that the value of the failure rate parameter increases with the increasing iteration times. It indicates that the proportion of sampling the paths satisfying the rare attribute is gradually increasing.

Figure 2 illustrates the distribution of the estimated values of the CESVA method during the iterations. The estimated value gradually converges from the 17th iteration. Figure 3 presents the distribution of the relative error of the CESVA method during the iterations. The relative error gradually converges from the 16th iteration. Finally, the probability estimated value of the security attribute φ is 1.682×10^{-12}, and the value of the relative error is 0.01.

In order to verify the statistical performance of the CESVA method, 100 experiments were carried out under the above parameters, and 2.0×10^5 samples were used in each experiment. Compared with the performance of the HIS method under the same sample size, Table 1 shows the mean, skewness, and statistical indicators such as standard deviation (likelihood ratio standard deviation), relative error, and sample size for each experiment. As presented in Table 1, with the same sample size, the estimated values of the CESVA method are more closely distributed around the mean value, and the likelihood is over 10 times less than the standard deviation and relative error, when comparing against the HIS method. Although the true probability is unknown, statistical indicators such as the standard deviation, skewness, and relative error of the likelihood ratio illustrate that the true probability and the mean are very close.

5. Related Work

The verification of the rare attribute for SMC mainly includes the importance sampling method, the importance splitting method, and the statistical learning method.

The importance sampling method is an effective method to solve the verification of rare attributes. For the CTMC and DTMC random models, Reijsbergen et al. [40] and Barbot et al. [23] leveraged the heuristic methods to obtain an importance sampling distribution to complete the attribute verification of the two types of models. For the Stateflow/Simulink model, Clarke and Zuliani [24] proposed the SMC method of cross-entropy minimization importance sampling to verify its safety properties. Zuliani et al. [17] further used the SMC method in paper [24] to verify the safety properties of a class of discrete-time SHS. The method proposed by Clarke and Zuliani [24] assumes that the distribution of the system path space is exponential distribution. By simply increasing the failure rate of the system parameters and calculating the optimal parameters of the exponential distribution with the paths satisfying the rare attributes extracted at one time, an importance sampling distribution can be obtained. J´egourel et al. [26] used a random guardian command to the importance sampling distribution. This model can approximate the path distribution of the system by increasing the number of commands (the number of parameters) and uses the minimized cross-entropy method to obtain an importance sampling distribution in the random model. However, the optimal importance sampling distribution obtained by the above method does not come from the distribution family of the system path space, and these methods actually belong to the heuristic importance sampling method.

The importance segmentation method [34] is a method of reducing the estimated variance. Based on the importance segmentation method, J´egourel et al. [33] proposed the SMC algorithm for the verification of small probability events. The key idea is to decompose the system logic

```
Input: $N$, the number of samples per iteration.
Input: $v$, the true path distribution parameter of SCCPS.
Input: $p^{(0)}$, the initialization parameter.
Input: $j_{\max}$, the maximum number of iterations.
Output: $p^*$ Optimal parameters.
Function learningAlg ($N$, $v$, $p^{(0)} j_{\max}$)
  $j = 0$;
    while $j < j_{\max}$ do
      $A = 0,\ B = 0,\ i = 1$
      while $i \leq N$ do
        generate a path $\omega_i$ according to the pdf $f(., p^{(j)})$
        if $\omega_i \vDash \varphi$ then
        $W_i = \sum_{\langle l,m\rangle \in S(\omega_i)} (v_{lm}/p_{lm})^{n_{lm}^{(\omega_i)}}$;
        $A = A + W_i n_{lm}^{(\omega_i)}$;
        $B = B + W_i n_l^{(\omega_i)}$;
      $i = i + 1$;
      $p_{lm}^{(j+1)} = \alpha p_{lm}^{(j)} + (1 - \alpha)(A/B)$;
    $j = j + 1$
  return $p^{(j-1)}$
```

Algorithm 1: Importance sampling distribution learning algorithm.

```
Input: $N_I S$, The number of samples.
Input: $v$, the true path distribution parameter of SCCPS.
Input: $p^*$, the optimal parameters calculated by Algorithm 1.
Output: $\widehat{p}$, Probability of SCCPS meeting safety attributes.
Function verifyingAlg ($N$, $v$, $p^{(0)} j_{\max}$)
  $A = 0,\ i = 1$
  while $i \leq N$ do
    generate a path $\omega_i$ according to the pdf $f(., p^{(j)})$
    if $\omega_i \vDash \varphi$ then
      $W_i = \sum_{\langle l,m\rangle \in S(\omega_i)} (v_{lm}/p_{lm})^{n_{lm}^{(\omega_i)}}$:
      $A = A + W_i$;
$i = i + 1$
return $(A/N_{IS})$
```

Algorithm 2: Safety verification algorithm.

Table 1: Comparison of statistical performance between CESVA and HIS.

Algorithm	Mean	Skewness	Standard deviation	Relative error
CESVA	1.687×10^{-12}	0.029	1.853×10^{-14}	0.011
HIS	1.986×10^{-12}	1.264	2.654×10^{-13}	0.133

attributes into embedded attributes, which makes its probability easier to be estimated and reduces the number of sample paths required by verification. To improve the performance, the attributes need to be decomposed into multiple levels with different probabilities. During the decomposition process, copying or eliminating paths depend on their intermediate behavior. When the decomposition is over, an estimated probability that the attribute is satisfied can be obtained. The importance segmentation method is essentially heuristic and depends on the model, but lacks the support of theoretical results.

Applying statistical learning methods to SMC is also an important research direction. Du et al. [19] proposed a learning SMC framework based on support vector machine-based two classifiers. It uses cost-sensitive and resampling methods to solve the unbalanced data learning problem of support vector machines and implements predicting and assessing the probability of occurrence of small-probability events with a relatively small number of samples. However, this method cannot obtain rare attribute samples. For the low-probability attributes of hardware circuits with multiple failure regions, Kumar et al. [41] assumed that the system failure distribution is a Gaussian mixture model, thus proposed to use the variational Bayes method to learn an optimal importance sampling distribution from the Gaussian mixture model. However, the optimal importance sampling distribution is not a distribution family from the system path space. Kalajdzic et al. [42] proposed an SMC method based on the principle of feedback control. This method learns a model of a cyber-physical fusion system by using importance sampling to estimate the system state and importance division to control the system. So it can infer the probability that the system satisfies the given attributes.

The method proposed in this paper starts from the SCCPS path probability space, constructs a cross-entropy optimization model, and uses an iterative learning method to obtain an optimal importance sampling distribution from the parameterized distribution clusters of the path space. It ensures that the optimal importance sampling distribution can come from the distribution family in the path probability space of SCCPS. And, the iterative learning method ensures that the distribution can evenly cover the unsafe path distribution area. Therefore, the accuracy and efficiency of the rare attribute verification can be improved significantly.

6. Conclusion

SMC has been successfully applied to SCCPS safety attribute verification and has become the most effective solution, but rare attribute verification is still a challenge for SMC. To be able to extract samples satisfying the rare attributes from SCCPS, CTMC is used to construct the probability space model of the execution path of SCCPS given with the probability measure of the random execution path as well as the parameterized probability distribution function family, to construct the cross-entropy iterative model. According to the iteratively learning from finding the approximate optimal importance sampling distribution in the SCCPS path probability space, the efficient sampling of rare attribute samples in SCCPS is achieved. With the evaluating experiments, the experimental results show that, for the verification of rare attributes, comparing against the heuristic importance sampling method with the same number of samples, the estimated value of our method is better distributed around the mean, and the standard deviation and relative error are reduced by more than an order of magnitude. Based on the method proposed in this paper, combining with the current mainstream SMC method to develop an adaptive SMC tool is set as the future work.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. The authors apply CESVA to a fault-tolerant controller for an aircraft elevator system (FTC4AE) that is a State-flow/Simulink hybrid system modeling case from MATLAB.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (no.2018YFB1003900), National Natural Science Foundation of China (no. 61772270), Key Laboratory of Safety-Critical Software (Nanjing University of Aeronautics and Astronautics), and Ministry of Industry and Information Technology Research Project (NJ2019006).

References

[1] N. A. Tanner, J. R. Wait, C. R. Farrar, and H. Sohn, "Structural health monitoring using modular wireless sensors," *Journal of Intelligent Material Systems and Structures*, vol. 14, no. 1, pp. 43–56, 2003.

[2] S. K. Kampf, M. Salazar, and S. W. Tyler, "Preliminary investigations of effluent drainage from mining heap leach facilities," *Vadose Zone Journal*, vol. 1, no. 1, pp. 186–196, 2002.

[3] G. Chunpeng, Z. Liu, J. Xia, and F. Liming, "Revocable identitybased broadcast proxy re-encryption for data sharing in clouds," *IEEE Transactions on Dependable and Secure Computing*, vol. 99, 2019.

[4] L. Yu and J.-p. Wang, "Review of the current and future technologies for video compression," *Journal of Zhejiang University Science C*, vol. 11, no. 1, pp. 1–13, 2010.

[5] H.-h. Xu and J. Zhu, "Aniterative approach to Bayes risk decoding and system combination," *Journal of Zhejiang University SCIENCE C*, vol. 12, no. 3, pp. 204–212, 2011.

[6] O. Déniz, M. Castrillón, J. Lorenzo, L. Antón, M. Hernandez, and G. Bueno, "Computer vision based eyewear selector," *Journal of Zhejiang University Science C*, vol. 11, no. 2, pp. 79–91, 2010.

[7] D. Theodoridis, Y. Boutalis, and M. Christodoulou, "Direct adaptive regulation of unknownnonlinear systems with analysis of themodel order problem," *Journal of Zhejiang University Science C*, vol. 12, no. 1, pp. 1–16, 2011.

[8] X.-c. Zhou, H.-b. Shen, and J.-p. Ye, "Integrating outlier filtering in large margin training," *Journal of Zhejiang University Science C*, vol. 12, no. 5, pp. 362–370, 2011.

[9] I. Prigogine, *Order through Fluctuation: Self-Organization and Social System*, pp. 93–134, Addison-Wesley, London, UK, 1976.

[10] C. Ge, W. Susilo, Z. Liu, J. Xia, P. Szalachowski, and F. Liming, "Secure keyword search and data sharing mechanism for cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 5, 2020.

[11] Z. Wu, Y. An, Z. Wang et al., "Study on zoelite enhanced contact-adsorption regeneration-stabilization process for nitrogen removal," *Journal of Hazardous Materials*, vol 156, 2008 in press.

[12] H. L. S. Younes, "Error control for probabilistic model checking," in *Lecture Notes in Computer Science*, E. A. Emerson and K. S. Namjoshi, Eds., pp. 142–156, Springer, Berlin, Germany, 2006.

[13] K. G. Larsen, "Statistical model checking, refinement checking, optimization, . . . for stochastic hybrid systems," in *Lecture Notes in Computer Science*, pp. 7–10, Springer, Berlin, Germany, 2012.

[14] Q. Wang, P. Zuliani, S. Kong, S. Gao, E. M. Clarke, and " SReach, "SReach: a probabilistic bounded delta-reachability analyzer for stochastic hybrid systems," *Computational Methods in Systems Biology*, vol. 9308, pp. 15–27, 2015.

[15] S. Gorini, M. Quirini, A. Menciassi, G. Permorio, C. Stefanini, and P. Dario, "A novel sma-based actuator for a legged endoscopic capsule," in *First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 443–449, Pisa, Italy, February 2006.

[16] U. Rizvi, *Combined Multiple Transmit Antennas and Multi-Level Modulation Techniques*, Stockholm, Sweden, Europe, in Swedish, 2006.

[17] P. Zuliani, C. Baier, and E. M. Clarke, "Rare-event verification for stochastic hybrid systems,," in *Proceedings of the ACM International Conference on Hybrid Systems: Computation & Control*, pp. 217–226, ACM, Quebec, Canada, April 2012.

[18] P. Zuliani, A. Platzer, and E. M. Clarke, "Bayesian statistical model checking with application to stateflow/simulink verification," *Formal Methods in System Design*, vol. 43, no. 2, pp. 338–367, 2013.

[19] D. Du, B. Cheng, and J. Liu, "Statistical model checking for rare-event in safety-critical system," *Journal of Software in Chinese*, vol. 26, no. 2, pp. 305–320, 2015.

[20] L. Sweeney, *Uniqueness of simple demographics in the U.S. population*, Technical Report No. LIDAP-WP4, Carnegie Mellon University, Pittsburgh, PA, USA, 2000.

[21] ISO, "Steels-classification-part 1: classification of steels into unalloyed and alloy steels based on chemical composition," Technical Report ISO 4948-1, ISO, Geneva, Switzerland, 1982.

[22] D. Reijsbergen, P. de Boer, W. R. W. Scheinhardt, and B. R. Haverkort, "Rare event simulation for highly dependable systems with fast repairs," in *Proceedings of the Seventh International Conference on the Quantitative Evaluation of Systems*, pp. 251–260, IEEE, Williamsburg, VA, USA, September 2010.

[23] B. Barbot, S. Haddad, and C. Picaronny, "Coupling and importance sampling for statistical model checking," *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 7214, pp. 331–346, 2012.

[24] E. M. Clarke and P. Zuliani, "Statistical model checking for cyber-physical systems," *Automated Technology for Verification and Analysis*, vol. 6996, pp. 1–12, 2011.

[25] University, *Citing Electronic Sources of Information*, University of Sheffield Library, Howard, UK, 2001, http://www.shef.ac.uk/library/libdocs/hsl-dvc1.pdf.

[26] C. J´egourel, A. Legay, and S. Sedwards, "Command-based importance sampling for statistical model checking," *Theoretical Computer Science*, vol. 649, pp. 1–24, 2016.

[27] H. L. S. Younes and R. G. Simmons, "Statistical probabilistic model checking with a focus on time-bounded properties," *Information and Computation*, vol. 204, no. 9, pp. 1368–1409, 2006.

[28] T. H´erault, R. Lassaigne, F. Magniette, and S. Peyronnet, "Approximate probabilistic model checking," in *Lecture Notes in Computer Science*, pp. 73–84, Springer, Berlin, Germany, 2004.

[29] Y. J. Kim, M. Kim, and T. Kim, "Statistical moHaifa, Israeldel checking for safety critical hybrid systems: an empirical evaluation," in *proceedings of the 8th international haifa verification conference on hardware and software: verification and testing*, pp. 162–177, Haifa, Israel, November 2012.

[30] G. Agha and K. Palmskog, "A survey of statistical model checking," *ACM Transactions on Modeling and Computer Simulation*, vol. 28, no. 1–6, pp. 6–39, 2018.

[31] A. Legay and M. Viswanathan, "Statistical model checking: challenges and perspectives," *International Journal on Software Tools for Technology Transfer*, vol. 17, no. 4, pp. 369–376, 2015.

[32] J. Hu, J. Lygeros, and S. Sastry, "Towards a theory of stochastic hybrid systems," *Hybrid Systems: Computation and Control*, vol. 337, pp. 160–173, 2000.

[33] C. J´egourel, A. Legay, and S. Sedwards, "An effective heuristic for adaptive importance splitting in statistical model checking," in *Lecture Notes in Computer Science*, pp. 143–159, Springer, Berlin, Germany, 2014.

[34] G. Jiang and M. C. Fu, "Importance splitting for finite-time rare event simulation," *IEEE Transactions on Automatic Control*, vol. 63, no. 6, pp. 1670–1677, 2018.

[35] D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook of monte carlo methods*, John Wiley & Sons, Hoboken, NJ, USA, 2013.

[36] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[37] A. Costa, O. D. Jones, and D. Kroese, "Convergence properties of the cross-entropy method for discrete optimization," *Operations Research Letters*, vol. 35, no. 5, pp. 573–580, 2007.

[38] B. Boyer, K. Corre, A. Legay, and S. Sedwards, "PLASMA-lab: a flexible, distributable statistical model checking library," in *Proceedings of the 10th International Conference on Quantitative Evaluation of Systems*, pp. 160–164, Buenos Aires, Argentina, August 2013.

[39] M. V. Stringfellow, N. G. Leveson, and B. D. Owens, "Safety-driven design for software-intensive aerospace and automotive systems," *Proceedings of the IEEE*, vol. 98, no. 4, pp. 515–525, 2010.

[40] D. Reijsbergen, P. de Boer, W. R. W. Scheinhardt, and B. R. Haverkort, "Rare event simulation for highly dependable systems with fast repairs," *Perform. Evaluation*, vol. 69, no. 7-8, pp. 336–355, 2012.

[41] J. A. Kumar, S. N. Ahmadyan, and S. Vasudevan, "Efficient statistical model checking of hardware circuits with multiple failure regions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 6, pp. 945–958, 2014.

[42] K. Kalajdzic, C. J´egourel, A. Lukina et al., "Feedback control for statistical model checking of cyber-physical systems," in *Proceedings of the leveraging applications of FormalMethods, verification and Validation: foundational techniques - 7th international Symposium, ISoLA 2016*, Imperial, Corfu, Greece, October 2016.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9961342, 14 pages
https://doi.org/10.1155/2021/9961342

Research Article

A Hierarchical Approach for Advanced Persistent Threat Detection with Attention-Based Graph Neural Networks

Zitong Li,[1] Xiang Cheng,[1] Lixiao Sun,[1] Ji Zhang,[2] and Bing Chen [1]

[1] *College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 21106, China*
[2] *School of Sciences, University of Southern Queensland, Toowoomba 4350, Australia*

Correspondence should be addressed to Bing Chen; cb_china@nuaa.edu.cn

Received 11 March 2021; Revised 15 April 2021; Accepted 26 April 2021; Published 4 May 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Zitong Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Advanced Persistent Threats (APTs) are the most sophisticated attacks for modern information systems. Currently, more and more researchers begin to focus on graph-based anomaly detection methods that leverage graph data to model normal behaviors and detect outliers for defending against APTs. However, previous studies of provenance graphs mainly concentrate on system calls, leading to difficulties in modeling network behaviors. Coarse-grained correlation graphs depend on handcrafted graph construction rules and, thus, cannot adequately explore log node attributes. Besides, the traditional Graph Neural Networks (GNNs) fail to consider meaningful edge features and are difficult to perform heterogeneous graphs embedding. To overcome the limitations of the existing approaches, we present a hierarchical approach for APT detection with novel attention-based GNNs. We propose a metapath aggregated GNN for provenance graph embedding and an edge enhanced GNN for host interactive graph embedding; thus, APT behaviors can be captured at both the system and network levels. A novel enhancement mechanism is also introduced to dynamically update the detection model in the hierarchical detection framework. Evaluations show that the proposed method outperforms the state-of-the-art baselines in APT detection.

1. Introduction

Advanced Persistent Threats (APTs) are becoming increasingly prominent in modern networks [1, 2]. Unlike conventional attacks, APTs are a class of sophisticated attacks launched by resourceful adversaries using a wide spectrum of attack techniques and tools [3]. The APT perpetrators initially compromise hosts or servers in a target environment and then stealthily traverse from system to system for internal reconnaissance and data breach [4, 5].

Traditional detection systems are insufficient to defend against APT attacks. Misuse-based detectors [6, 7] that learn patterns associated with known attacks are difficult to detect APTs because of their nonrepetitive behaviors. In contrast, anomaly-based detectors [8–13] are capable of identifying unforeseen activities that do not conform to the learned normal patterns. However, they are susceptible to be circumvented by attackers because they typically treat system calls or network events as temporal sequences [8, 9, 11], which only carry the sequential relationships among log entries. As such, they cannot achieve satisfactory performance in detecting APTs [1].

Recent works suggested that the provenance graph is a better tool for threat modeling and APT detection [14]. Provenance graphs represent system executions as control flows and data flows between subjects and objects. The naïve method is to perform rule-based subgraph matching for APT detection [1, 15, 16], but it is incapable of dealing with unknown APT patterns. Moreover, provenance graphs only focus on system-level information and thus cannot effectively model network-level behaviors. Besides, the correlation graphs [3, 17] treat log entries as nodes and bridge them by rules. The limitation of log graphs is that they require too much expert knowledge to define proper correlation rules, and coarse-grained log nodes ignore semantic information of log attributes, making it difficult for them to capture system-level APT patterns.

Traditional graph embedding methods, including DeepWalk [18] and node2vec [19], rely on heuristic algorithms to aggregate graph structural information. These methods are inherently transductive and fail to encode node attributes for graph embedding. Graph Neural networks (GNNs) [20–22] are deep learning methods to perform graph representation learning with good scalability and generalization ability. A series of GNN-based anomaly detection systems [12, 23, 24] have been proposed to perform inductive graph embedding by learning a set of aggregator functions to aggregate neighbor's features. Unfortunately, these models are not well-suited for APT detection. Because it is difficult for them to encode heterogeneous graphs integrating multiple types of system entities and operations, and most of them discard meaningful edge attributes that represent interactions between nodes.

To overcome the limitations of existing approaches, we propose a hierarchical approach that is capable of effectively detecting APTs with novel attention-based GNNs. Our approach comprises three components as shown in Figure 1. (1) Graph construction: we construct the Intrahost Provenance Graph (IPG) and the Interhost Interactive Graph (IIG) to comprehensively capture the behaviors of the full APT lifecycle. (2) Graph embedding and detection algorithm: for the IPG, we propose a metapath aggregated GNN and train an autoencoder to identify anomalous hosts; for the IIG, we present an edge feature-enhanced GNN and leverage the negative sampling to detect anomalous interactions among hosts. (3) Anomaly detection and model update: after the anomalous hosts and events are detected, the enhancement mechanism dynamically updates the IIG detection model using the reported malicious hosts from the IPG detectors.

We introduce specialized designs to tackle the aforementioned problems. First, the IPG and the IIG enable modeling behaviors both at system and network levels. Second, we encode the heterogeneous IPG using the metapath aggregated GNN with the attention mechanism, which enables the full exploration of semantic information using metapaths tailored to the IPG. Third, for the IIG, the interaction edges among hosts contain meaningful information; thus, we propose the edge feature-enhanced GNN to adequately exploit the multidimensional edge features. Additionally, with the enhancement from the IPG detectors, the IIG detection model can be efficiently updated to learn new patterns, and the malicious behaviors among hosts can be further detected after the compromised hosts are reported. Finally, we introduce a compact three-stage APT model and map the anomalies at both the system and network levels to the corresponding APT stages according to the hierarchical detection framework.

Specifically, the contributions of this article are summarized as follows:

(i) We propose a novel metapath aggregated GNN that models complex semantic and structural information of the system-level provenance graph.

(ii) We present a novel edge feature-enhanced GNN that models rich interactive information of the network-level host interactive graph.

(iii) We introduce an enhancement mechanism to dynamically update the network-level IIG model using the reported malicious host from the system-level IPG detectors to learn new anomalous patterns over time.

(iv) We propose a compact three-stage APT model and a hierarchical detection framework where the system-level and network-level anomalies can be mapped to the corresponding APT stages.

(v) The proposed method is evaluated on the StreamSpot and the LANL datasets for system-level and network-level APT attack detection. Experimental results show that our method outperforms the state-of-the-art approaches.

The remainder of the article is organized as follows. Section 2 reviews the related works. In Section 3, we present the schematic architecture, the compact three-stage APT model, and the hierarchical detection framework. The graph construction is shown in Section 4. Section 5 presents the details of the anomaly detection in provenance graphs. Section 6 presents the network-level anomaly detector and the model update mechanism. Evaluation results are presented and discussed in Section 7. The limitations and future works are discussed in Section 8. The conclusion is drawn in Section 9.

2. Related Work

Our work lies in the intersection of log analysis, provenance-based threat modeling, and graph-based intrusion detection. Therefore, we discuss the existing works in the following related areas. The taxonomy and representative publications are shown in Table 1.

2.1. Sequence-Based Log Analysis. Modern systems constantly generate logs and events that describe system status at various critical points, which are ideal sources of information for attack detection and system failures debugging [32]. LogLens [11] is a real-time anomaly detection system that deployed an unsupervised learning method to analyze log sequences without the knowledge of target systems and user specifications. Advances in natural language processing have shed light on log analysis. DeepLog [8] converts system logs into natural language sequences by utilizing the Long Short-Term Memory (LSTM) network, which can automatically learn normal patterns from system behaviors and alert for anomalies. To identify rare anomalies in the constantly evolving environment, Parveen [9] presented an unsupervised ensemble learning algorithm that compresses repetitive sequences to a dictionary to detect anomalies. In addition to detecting malicious activities as they happen, attack prediction is still an open research problem. Tiresias [25] leverages the Recurrent Neural Network (RNN) to predict specific attack steps by considering previously observed events.

These log analysis approaches mostly treat logs as temporal sequences, which only hold sequential

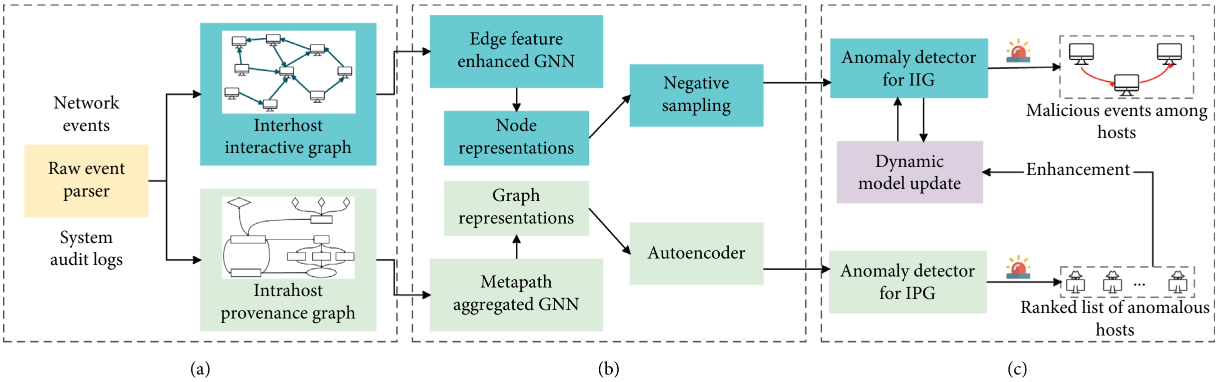


Figure 1: Schematic architecture: (a) graph construction; (b) graph embedding and detection algorithm; (c) anomaly detection and model update.

relationships among log entries. Therefore, they ignore the semantic relationships among system entities and the interactive relationships among hosts, whereas our method considers these relationships for anomaly detection.

2.2. Provenance-Based Threat Modeling. More and more research works start to focus on the provenance graphs that model the control flow and data flow between system-level entities [14]. Provenance is typically used for forensic analysis and attack attribution. BackTracker [28] leverages the provenance tracking system to identify the entry point of attacks, while PriorTracker [29] enhances it by timely forward causality tracking and automatically prioritizing the abnormal causal dependencies. To combat the challenge named “threat alert fatigue,” NoDoze [30] uses contextual and historical information of alerts in the provenance dependency graph and performs attack triage by identifying anomalous subgraphs. SPADE [26] is an open-source provenance collection and management framework, which decouples the function of collection, storage, and querying. Pasquier et al. [27] designed a practical provenance capture system, called CamFlow, that can tailor the captured data to reduce the overhead.

The increasingly sophisticated APTs prompt a number of researchers to use provenance graphs for APT analysis. SLEUTH [15] constructs memory-based provenance graphs that significantly improve the speed of data processing and uses the trustworthiness and confidentiality tags for code and data to perform source identification and impact analysis. HOLMES [1] uses the semantic information of provenance to construct a customized policy framework, and the behaviors that conform to the policies are further mapped to Tactics, Techniques, and Procedures (TTPs), which are eventually mapped to high-level APT kill-chain stages. Poirot [16] uses provenance graphs to perform threat detection. It extracts Indicators of Compromise (IoCs) from Cyber Threat Intelligence (CTI) related to APTs to construct the query graphs and aligns them with provenance graphs constructed out of kernel audit logs to detect attack behaviors. However, these rule-based methods require prior expert knowledge of known APT patterns and thus cannot deal with unknown APT attacks. Our method instead is an anomaly-based system that requires no expert knowledge or labeled anomalous data.

2.3. Graph-Based Anomaly Detection. Anomaly detection with graph data has been widely researched and applied to identify outliers [12]. To analyze temporal evolution of insider threat events, Moriano et al. [31] proposed an unsupervised learning method to capture interactions between users and systems by constructing a bipartite graph. Log2vec [3] optimizes the correlated log graphs by constructing a rule-based heterogeneous graph integrating multirelationships among log entries. Then, Log2vec presents an improved graph embedding algorithm based on the random walk and word2vec and leverages the clustering threshold detector to identify malicious events. StreamSpot [13] is a memory-efficient anomaly detection system that deals with provenance graph heterogeneity and streaming challenge. To mitigate the drawbacks of StreamSpot in handling locally constrained graph features and dynamically clusters maintaining, UNICORN [2] analyzes contextualized provenance graphs to detect APTs. It is capable of modeling and summarizing the evolving system executions and report anomalous system status.

However, these approaches focus on either the system operations or the network flows for graph analysis, making them unable to fully capture the activities of APT attacks. Also, they depend on handcrafted algorithms that are difficult to generalize to different settings. GNNs have been widely adopted to learn node embeddings in an unsupervised way for anomaly detection. Ding et al. [23] studied the graph anomaly detection problem and deployed a novel model with the synergy of Graph Convolutional Network (GCN) and autoencoder. AddGraph [24] further combines GCN with Gated Recurrent Unit (GRU) to capture the long-term and short-term temporal patterns using an attention model. However, these GNN-based anomaly detection methods are not well-suited to APT detection because they either discard meaningful edge features or difficult to embed heterogeneous graphs. We introduce the metapath aggregated GNN to fully explore the semantic information embedded in the heterogeneous IPG, and we proposed the edge

Table 1: Taxonomy and representative publications of the related works.

Category		Publications
Sequence-based log analysis	Attack detection	[8, 9, 11]
	Attack prediction	[25]
Provenance-based threat modeling	Provenance capture	[26, 27]
	Forensic analysis	[28–30]
	Rule-based APT detection	[1, 15, 16]
Graph-based anomaly detection	Provenance graph	[2, 13]
	Correlation graph	[3, 31]
	GNN-based methods	[23, 24]

feature-enhanced GNN to adequately exploit the edge attributes in the IIG.

3. Overview

3.1. Architecture. As shown in Figure 1, our approach is composed of three key components:

3.1.1. Graph Construction. To capture all footprints left by APT attackers, we collect multimodal information of the system audit logs from different operating systems and network events from various protocols. The input to the graph construction component is the raw network events and system audit logs, which are normalized to uniform formats by the data parser. System audit records from different OS platforms are normalized to a common data representation of system entities and operations. Network events involving different protocols are normalized to a common data representation of interactions between hosts.

To capture the structural information and semantic dependencies, the parsed system audit logs are used to construct the IPG where all system entities are treated as nodes and operations are treated as edges, and the parsed network events are used to construct the IIG where all hosts are treated as nodes and events among them are treated as edges. By doing so, the execution status of the target environment is comprehensively captured from both system-level and network-level graphs. The IPG is capable of modeling system-level APT behaviors, such as exploitation of target systems and malicious code execution. The network-level APT events, which can be captured by the IIG, are not limited to lateral movement and DNS communication with Command and Control (C & C) servers.

3.1.2. Graph Embedding and Detection Algorithm. Graph embedding, also known as graph representation learning, is an approach that is capable of transforming graph nodes into vectors with a lower dimension while maximally preserving structural and semantic information. Specifically, we proposed a metapath aggregated GNN for the IPG and an edge feature-enhanced GNN for the IIG. For the system-level IPG with multitypes of entities and operations, we aggregate every single metapath instance, including the neighbor nodes and intermediate nodes, to a latent representation vector; then, we combine them by an attention-based mechanism to obtain the embedding of target nodes. For the network-level IIG with rich edge features, we leverage the novel edge feature-enhanced embedding algorithm based on the attention mechanism to adequately incorporate the interactions among host nodes.

After converting nodes into latent vectors, we design two detection algorithms for anomaly detection at both the system and network levels. Note that we have no labeled anomaly data to training the detection model; thus, the detection models are trained in an unsupervised manner. For the IPG, we use graph embeddings to train an autoencoder model by optimizing the reconstruction errors that are used to flag anomalies. Moreover, for the IIG, we compute the anomaly scores for edges based on the corresponding node embeddings and leverage the negative sampling to generate anomalous edges for training the detection model.

3.1.3. Anomaly Detection and Model Update. At last, the learned models are implemented to identify anomalous network events and system operations. The IPG detectors report a ranked list of suspicious hosts that perform system-level malicious actions inside, while the IIG detector reports a set of malicious events among hosts. However, the training data recording normal executions are usually incomplete and noisy, which may lead to an excessive number of false alarms during anomaly detection. To enhance the detection model, we use the reported suspicious hosts from the IPG detectors to dynamically update the IIG detection model. Specifically, after the IPG detectors report a list of suspicious hosts, malicious interactions appear among these hosts with high probability. Thus, we treat the interactions among these hosts as malicious samples to dynamically update the IIG detection model to learn new anomalous patterns over time.

3.2. Three-Stage APT Model. In contrast to other conventional attacks, APTs are characterized by persistence, diverse attack vectors, and low-and-slow attack patterns, which give rise to multiple attack stages of the APT lifecycle. MITRE ATT & CK framework presents a 14-stage APT knowledge base to describe adversary tactics and techniques of APT attacks. The Lockheed Martin cyber kill-chain [33] is a 7-phase framework to describe the sophisticated APT attack process. Nevertheless, some stages, such as the Initial Reconnaissance, are difficult to be detected given that the actions of the phase are conducted without any interaction with the target network. As such, it is impracticable to design a comprehensive approach to detect all the stages of the APT lifecycle. Moreover, an overly redundant multistage model can only be used to better for understanding the evolution of APTs, but not for detecting them.

After analyzing various APT cases, a compact three-stage APT model that is at the heart of the APT lifecycle is proposed. The invariant parts of APTs include three stages:

(i) Infiltration and malicious code execution: at first, the attacker must deploy the malware and execute the malicious code in the victim host to exploit an entry point of the target network or establish a foothold for the next move.

(ii) Internal reconnaissance and lateral movement: the goal of the APT perpetrator is to steal confidential data or to damage critical network components. To this end, the attacker would need to further compromise more vulnerable hosts, traverse from system to system within the target environment, and escalate privileges at any time if needed.

(iii) C & C communication and data exfiltration: the infected host would try to communicate with the C & C server to receive remote attack instructions or any other relevant tasks. Besides, actions that involve exfiltrating data to the C & C server and undermining critical components fall under this stage.

The compact three-stage APT lifecycle is presented to model APT attacks performed in complex networks and is applicable and necessary for most APT scenarios.

3.3. The Hierarchical APT Detection Framework. As shown in Figure 2, the hierarchical APT detection framework maps anomalies of system-level hosts and network-level events to high-level APT stages. APT attackers not only deploy malicious code inside specific hosts to manipulate processes and files but also move laterally for internal reconnaissance and information exfiltration. The IPG detectors over the Intrahost Provenance Graphs report malicious hosts that are suffering attacks. Further, these hosts detected at the system layer can be used as seeds to analyze which internal hosts they try to communicate with at the network layer. Moreover, the IIG detection model can also be dynamically updated by learning detection results from the IPG detectors. At last, the infected host nodes and attack paths among them are mapped to a compact three-stage APT model described above. Bringing together the information of system audit logs and network events into graph data allows us to comprehensively model the status of the target environment, and the synergy between the IPG and IIG detection model enables our method to capture footprints of APT attacks by considering the target environment as a whole.

4. Graph Construction

4.1. Intrahost Provenance Graph. The system-level provenance graph enables a strong semantic expression of system execution and is increasingly attractive for researchers in the APT analysis. The provenance collection system captures the record information at the system level from different operating systems; thus, causality dependencies can be properly captured. As shown in Figure 3, the provenance graph treats all subjects (e.g., processes, threads) and objects (e.g., files, sockets) as nodes and all operations from subjects to objects (such as a process reading a file, interprocess communication, and a process sending data to an Internet socket) as directed edges. The built-in auditing systems, such as Windows ETW (Event Tracing for Windows) and Linux Auditd, provide coarse-grained provenance capture, while the extra provenance infrastructure, such as Hi-Fi [34], LPM [35], and CamFlow [27], are used to collect more fine-grained provenance.

Following the idea of the provenance graph, we collect system operations from all hosts of the target network and model these event data as a platform-neutral heterogeneous graph for each host. Formally, a heterogeneous IPG is denoted by IPG = (V, E), in which V is a set of nodes involving subjects and objects and E denotes a set of events. The types of subject and object in IPG include process, thread, file, and socket. It is noteworthy mentioning that the subject and object are relative, meaning that a subject of an event can be the object of another operation. The event in IPG can be represented by a 4-tuple <subject, object, operation, *timestamp*> where the operation types include the fork, clone, open, write, read, etc. The IPG is used to detect the infiltration and malicious code execution of the APT model.

4.2. Interhost Interaction Graph. To exfiltrate confidential data or sabotage critical components, the APT perpetrators would need to laterally move across multiple hosts of the target Intranet to escalate privilege and search for the components and data. Besides, the infected hosts would try to communicate with the C & C server via the DNS requests to receive the instructions and exfiltrate data. As such, the analysis of interactive relationships among multiple hosts is essential for the detection and mitigation of APT activities. To this end, the Interhost Interaction Graph is proposed to model the communication and connectivity of the entire target network.

As shown in Figure 4, in the IIG, each node represents a host or server while the interactions between them are denoted by edges. In an enterprise, hosts own different users who tend to conduct diverse operations with internal and external nodes. For instance, administrators often log into multiple hosts for policy management, parameter configuration, and crash recovery. Moreover, file transmission and domain name lookup are frequently performed, which could generate a volume of data flow and DNS traffic. To capture these interhost behaviors, the IIG takes network flow events, authentication events, and DNS lookup events into consideration. Formally, the Interhost Interaction Graph is denoted by IIG = (V, E), in which V is the set of host nodes and E is the set of events conducted by the source host to the destination host. The IIG is mainly used to detect the last two stages of the proposed APT model.

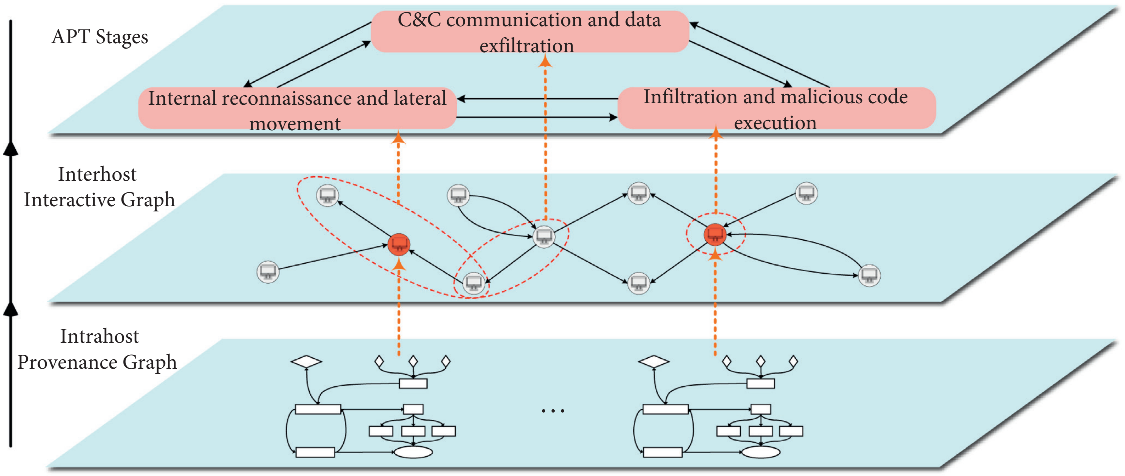

APT Stages
C&C communication and data exfiltration
Internal reconnaissance and lateral movement
Infiltration and malicious code execution
Interhost Interactive Graph
Intrahost Provenance Graph
...

Figure 2: The hierarchical APT detection framework.

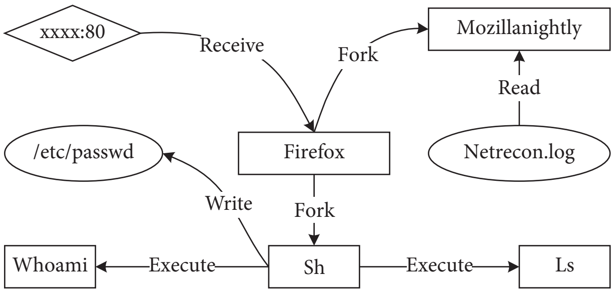


Figure 3: The intrahost provenance graph.

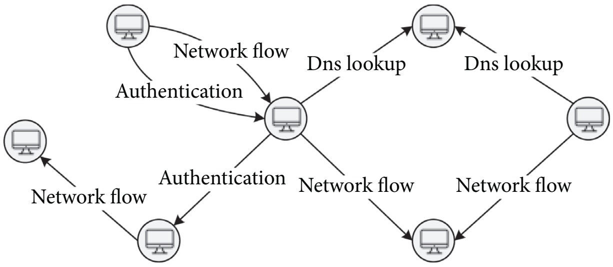


Figure 4: The interhost interaction graph.

5. Anomaly Dete ction on the IPG

5.1. Metapath Aggregated GNN. The constructed IPGs are heterogeneous with multiple different types of system entities and causal dependencies. Thus, we propose a novel metapath aggregated GNN to embed nodes of the heterogeneous IPG into low-dimensional node representations in a semantically meaningful way. The definitions of metapath and metapath instance are as follows:

> Metapath: Ametapath P is an ordered sequence in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$ defined on the graph network schema $T_{\mathscr{G}} = (\mathscr{A}, \mathscr{R})$, which describes a composite relation $R = R_1 \cdot R_2 \cdot \cdots \cdot R_l$ between node types A_l and A_{l+1}, where l denotes the length of P and $\cdot$ is the composite operator on relations. For simplicity, we use an abbreviation form $P = (A_1, A_2, \ldots, A_{l+1})$ to denote a metapath when there is only one event type between a pair of entities.
>
> Metapath instance: Given that metapath $P = (A_1, A_2, \ldots, A_{l+1})$, if $\forall i, \phi(v_i) = A_i$ and $e_i = <v_i, v_{i+1}>$ belongs to the relationship $R_i \in P$, then metapath instance p is presented in the form of $p = (v_1, v_2, \ldots, v_{l+1})$ following the schema defined by metapath P.

As shown in Figure 5, we propose a metapath encoder for the IPG embedding. Given metapath P, the metapath encoder combines the information embedded in the metapath context by aggregating the metapath-based neighbors of the target node v and the intermediate nodes for each metapath instance of P. Specifically, let $p(v,u)$ denote a metapath instance of P correlating the target node v and its metapath-based neighbor $u \in \mathscr{N}_v^P$, and $I_{p(v,u)} = p(v,u) \setminus \{v,u\}$ denotes the set of intermediate nodes within $p(v,u)$. The metapath encoder first employs a node aggregator to embed all node embeddings along with a metapath instance into a single vector:

$$\mathbf{h}_{p(v,u)} = f\left(\mathbf{h}'_v, \mathbf{h}'_u, \left\{\mathbf{h}'_t, \forall t \in I_{p(v,u)}\right\}\right), \qquad (1)$$

where $\mathbf{h}'_v$, $\mathbf{h}'_u$, and $\mathbf{h}'_t$ are the input node representations, $f(\cdot)$ is a node aggregation function; $\mathbf{h}_{p(v,u)}$ is the aggregated output of the metapath instance $p(v,u)$.

After aggregating every single metapath instance into a latent representation vector, the metapath encoder combines different instances of metapath P regarding target node v through the graph attention layer. It is reasonable to use the weighted sum of metapath instances because different instances would have different degrees of contribution to the target node's embedding. Thus, all metapath instances can be weighted summed by learning a normalized attention coefficient:

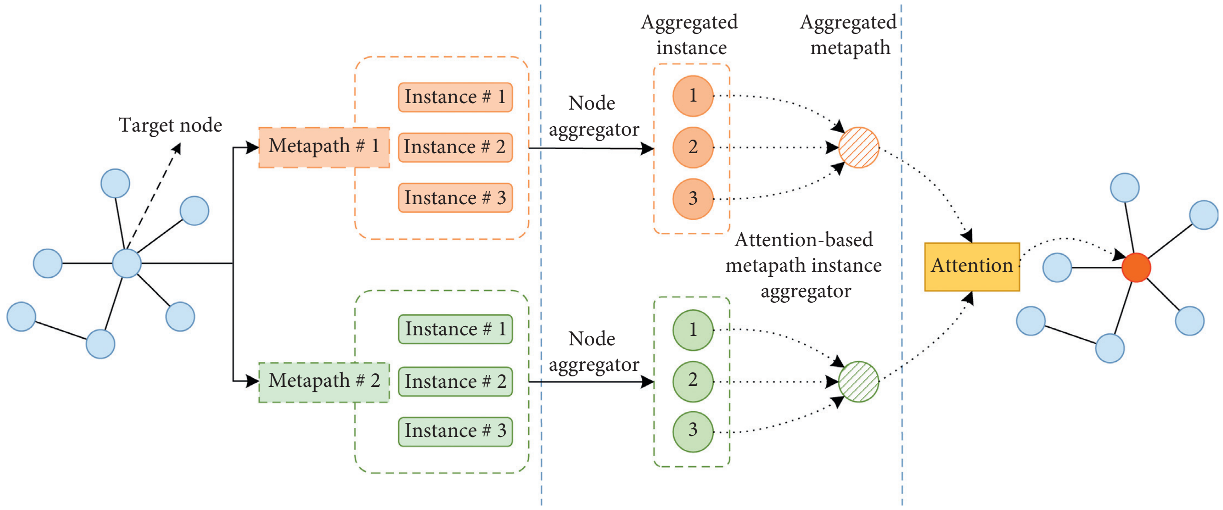


Figure 5: The metapath encoder for Intrahost Provenance Graphs.

$$e^P_{vu} = a\left(\mathbf{Wh}'_v, \mathbf{Wh}_{p(v,u)}\right),$$

$$\alpha^P_{vu} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}_P^\top\left[\mathbf{Wh}'_v \| \mathbf{Wh}_{p(v,u)}\right]\right)\right)}{\sum_{s\in\mathcal{N}_v^P}\exp\left(\text{LeakyReLU}\left(\mathbf{a}_P^\top\left[\mathbf{Wh}'_v \| \mathbf{Wh}_{p(v,s)}\right]\right)\right)}, \quad (2)$$

$$\mathbf{h}_v^P = \sigma\left(\sum_{u\in\mathcal{N}_v^P}\alpha^P_{vu}\cdot\mathbf{h}_{p(v,u)}\right),$$

Here, $\mathbf{a}_P$ is a weight attention vector for metapath P. We concatenate the representations of target node v and metapath instance embedding $p(v,u)$ parametrized by $\mathbf{a}_P^\top$ and apply the LeakyReLU nonlinear activation function to compute the attention coefficient e^P_{vu}, which indicates the importance of the representation of the metapath instance $p(v,u)$ to the node v. The coefficients are normalized to make them easily comparable across different instances using the softmax function, and then they are used as the weights to compute the combination of all representations of different metapath instances about node v. The output finally goes through an activation function $\sigma(\cdot)$ to obtain the representation of target node v regarding metapath P.

After aggregating the information contained in each metapath, we need to further aggregate the structural and semantic information among all different metapaths. For a given target node v and the set of metapaths $\mathcal{P} = \{P_1, P_2, \ldots, P_K\}$, we now have K representations for each metapath P_i denoted as $\left\{\mathbf{h}_v^{P_1}, \mathbf{h}_v^{P_2}, \ldots, \mathbf{h}_v^{P_K}\right\}$. The metapath encoder further employs an attention mechanism to combine the metapath-specific representations by assigning different weights to different metapaths. First, we average the transformed representations of metapath $P_i \in \mathcal{P}$ for all nodes as

$$\mathbf{e}_{P_i} = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\tanh\left(\mathbf{W}\cdot\mathbf{h}_v^{P_i} + \mathbf{b}\right), \quad (3)$$

where $\mathbf{W}$ and $\mathbf{b}$ are the weight and bias parameters. Then, the attention-based mechanism is used to aggregate the metapath-specific representations of the target node v as follows:

$$\alpha_{P_i} = \frac{\exp\left(\mathbf{a}^\top\cdot\mathbf{e}_{P_i}\right)}{\sum_{P_j\in\mathcal{P}}\exp\left(\mathbf{a}^\top\cdot\mathbf{e}_{P_j}\right)},$$
$$\mathbf{h}_v = \sum_{P_i\in\mathcal{P}}\alpha_{P_i}\cdot\mathbf{h}_v^{P_i}, \quad (4)$$

where $\mathbf{a}$ is a weight attention vector and α_{P_i} indicates the relative importance of metapath P_i to the target node v. Once $\mathbf{e}_{P_i}$ is obtained for each metapath $P_i \in \mathcal{P}$, we use the normalized importance α_{P_i} to sum up all metapath-specific representations of node v in a weighted way to obtain the final node representation $\mathbf{h}_v$. Finally, to obtain the graph level representation $\mathbf{h}_g$, the mean readout operation is performed by averaging all the representations of the target nodes.

5.2. Autoencoder-Based Anomaly Detection. To detect anomalies launched by APT attackers, an anomaly detection mechanism based on the deep autoencoder is designed to reconstruct the learned graph embeddings. The deep autoencoder is used to learn data encodings in an unsupervised fashion, typically for data denoising and dimensionality reduction. Here, we leverage it to perform anomaly detection by computing the reconstruction errors that are used to flag anomalies. The intuition behind this is that the instances with large reconstruction errors are more likely to be anomalies because their behavior patterns significantly deviate from the normal patterns and thus cannot be effectively reconstructed from the observed data. The deep autoencoder is capable of identifying those hosts that do not conform to the expected normal patterns.

For system-level anomaly detection, the autoencoder takes the representations of IPG snapshots as its inputs. Thus, the anomaly detection in the provenance graph scenario can be simply regarded as a graph classification problem. Specifically, given graph embedding $\mathbf{h}_g$, the encoder is first implemented to compress the input into a latent feature space with a lower dimension. After that, the decoder attempts to reconstruct the original data based on the latent vector. We can optimize the model parameters by minimizing the following MSE loss function:

$$\mathscr{L}_{\mathrm{AE}} = \frac{1}{n} \sum_{i=1}^{n} \left\| \mathrm{Dec}\left(\mathrm{Enc}\left(\mathbf{h}_g\right)\right) - \mathbf{h}_g \right\|^2. \tag{5}$$

The loss function takes ℓ_2-norm distance as the measurement of the reconstruction errors. After several training iterations, the anomaly score of each IPG snapshot can be computed as follows:

$$\mathrm{S}\left(\mathbf{h}_{\mathrm{g}}\right) = \left\| \mathrm{Dec}\left(\mathrm{Enc}\left(\mathbf{h}_g\right)\right) - \mathbf{h}_g \right\|^2. \tag{6}$$

It indicates that the snapshots with large anomaly scores are more likely to be under attack. Thus, we can rank the top-k suspicious hosts and set threshold λ to trigger alerts.

6. Anomaly Detection on the IIG

6.1. Edge Feature-Enhanced GNN. The Interhost Interaction Graph is a combination of host nodes and network-level information flows over the whole target network. For the IIG, the host nodes are relatively uniform, whereas the interactions between them vary in types, including authentication, network flow, and DNS lookup. These events contain important information about the network activities. Consequently, multidimensional edge features of IIG events should be further exploited to capture the interactive information. We propose a novel edge feature-enhanced GNN that leverages the attention mechanism to fully incorporate edge features for the IIG representations.

Given an Interhost Interaction Graph, let $\mathbf{E}_{vu}$ be a tensor of edge features between nodes v and u, and $\mathbf{E}_{vu}^{p}$ is the p^{th} channel of the edge feature in P-dimensional feature vector $\mathbf{E}_{vu}$. In contrast to the existing attention mechanism described in GAT [22], the proposed mechanism allows us to implement attention operations guided by the edge features. For the IIG with various activities, we consider the multidimensional edge features as multichannel signals, and each channel of signals can guide an independent attention operation, respectively. For a specific channel for the p^{th} edge feature, the normalized attention coefficient is computed as follows:

$$\begin{aligned} e_{vu}^{p} &= f\left(\mathbf{h}_v', \mathbf{h}_{\mathrm{u}}'\right)\mathbf{E}_{vu}^{p}, \\ a_{vu}^{p} &= \frac{\exp\left(\mathrm{LeakyReLU}\left(e_{vu}^{p}\right)\right)}{\sum_{s\in\mathscr{N}_v}\exp\left(\mathrm{LeakyReLU}\left(e_{vs}^{p}\right)\right)}, \end{aligned} \tag{7}$$

where $\mathbf{E}_{vu}^{p}$ is the p^{th} channel feature of the edge connecting nodes v and u, and $f(\cdot)$ is an arbitrary attention function that produces a scalar importance value from two input embedding vectors. Here, we adopt a linear function to perform the following:

$$f\left(\mathbf{h}_v', \mathbf{h}_u'\right) = \mathbf{a}^{\top}\left(\mathbf{W}\mathbf{h}_{\mathbf{v}}' \| \mathbf{W}\mathbf{h}_{\mathbf{u}}'\right). \tag{8}$$

At last, we aggregate the neighbor embeddings and the corresponding edge features based on the attention mechanism to generate the new embedding for the target node. The aggregation operation can be formulated as follows:

$$\mathbf{h}_v = \|_{p=1}^{P}\, \sigma\left(\sum_{u\in\mathscr{N}_v} \alpha_{vu}^{p}\mathbf{W}\left(\mathbf{h}_u', \mathbf{E}_{vu}^{p}\right)\right), \tag{9}$$

where $\mathbf{h}_v$ is the new embedding of the target node. We use weight α_{vu}^{p} to sum up neighbor embeddings and the edge feature of a specific channel. The results produced by all different channels of the edge features are combined by the concatenation operation. By doing so, the multidimensional edge features can be seamlessly aggregated into the node embeddings, which helps capture the structural and semantic information across the whole target network.

6.2. Negative Sampling. Due to the characteristics of APT attacks, it is difficult to collect sufficient labeled data that describe APT attack patterns comprehensively. So, it results in poor performance if we train a detection model in a supervised fashion that the labeled data cover only a small part of the possible anomalous operations. Besides, the goal of anomaly detection in IIG is to discriminate the anomalous edges when new events occur in the target network. Thus, we compute the anomaly scores for edges based on the corresponding node embeddings and leverage the negative sampling to train the detection model. Given an IIG snapshot G_t at timestamp t, we produce the vector representations of all host nodes. For each incoming edge (v, u) of G_t, the anomaly score can be computed as follows:

$$\mathrm{S}(v, u) = \omega \cdot \sigma\left(\beta \cdot \left(\left\|\mathbf{a}\odot\mathbf{h}_v + \mathbf{b}\odot\mathbf{h}_u\right\|_2^2 - \mu\right)\right), \tag{10}$$

where $\mathbf{h}_v$ and $\mathbf{h}_u$ denote the vector representations of node v and node u, respectively, and $\sigma(\cdot)$ is the sigmoid function. The node embeddings are parametrized by weight vectors $\mathbf{a}$ and $\mathbf{b}$ that are optimized in the output layer. β and μ represent hyperparameters in the anomaly score function.

To overcome the challenge of insufficient anomaly data, we leverage the negative sampling to train a detection model that learns the normal behaviors of the target network. Specifically, we generate a negative sample for each normal edge as an anomalous edge. The negative sampling means replacing those nodes at the end of normal edges with other nodes in the graph. For example, given normal edge (v, u), the set of selected nodes should be $\mathrm{V}\setminus\mathscr{N}_v$ if node u is replaced. Thus, the edges are assigned to the hosts that originally had no interactions. We define a Bernoulli distribution for the negative sampling: for normal edge (v, u), we corrupt the event by replacing the head host with a probability of $d_v/(d_v + d_u)$ or replacing the tail host with a probability of $d_u/(d_v + d_u)$, where d_v and d_u denote the degree of node v and node u, respectively.

It is worth noting that the generated negative samples may still be normal; thus, the strict loss functions, such as the cross-entropy, cannot be used to discriminate the original edges and the generated ones. Thus, we adopt the margin-based pairwise loss function in training the detection model:

$$\mathscr{L}^t = \min \sum_{(v,u)\in\mathscr{E}^t} \sum_{(v',u')\notin\mathscr{E}^t} \max\{0, \gamma + S(v,u) - S(v',u')\}, \tag{11}$$

where $s(v,u)$ and $s(v',u')$ are the anomaly scores of the existing edges and the generated edges, respectively. $\gamma \in (0,1)$ is the margin between the anomaly scores of the normal edge and the anomalous one. The minimization of the loss function $\mathscr{L}^t$ drives $s(v,u)$ to be smaller while $s(v',u')$ to be larger, which is consistent with our goal.

6.3. Dynamic Model Update. The detection models proposed in this article do not require any anomalous data for training, so they are capable of detecting unforeseen anomaly types such as zero-day attacks exploited in APT campaigns. However, the training data are usually noisy and incomplete, so false positives and false negatives may be generated when we use the trained model for anomaly detection. Note that the IPG describes normal patterns from the perspective of system events and the IIG models normal behaviors at the network level. The promising idea is to combine the IPG and IIG to build an anomaly detection mechanism that fully integrates both sources of information. To this end, we propose an enhancement mechanism that dynamically updates the IIG detection model using the reported malicious hosts by the IPG detectors.

Recall that the IPG anomaly detectors will report a set of hosts that do not conform to the learned normal patterns. Administrators can diagnose the systems by referring to the list of suspicious hosts. Beyond that, APT actors may further move laterally to infiltrate across multiple hosts after compromising a certain target. To detect the subsequent actions of APT attacks, we use the detected suspicious hosts to enhance the IIG detector by dynamically updating the detection model. Note that the false-positive samples may appear in the reported list of suspicious hosts; thus, before updating the IIG model with newly reported hosts from the IPG detectors, the security analysts should manually remove the false-positive samples to ensure that the hosts used for the update are all compromised.

We can update the IIG detection model by defining the loss function $\mathscr{L}^t_{\text{update}}$ as follows:

$$\mathscr{L}^t_{\text{update}} = \min \sum_{(v,u)\notin\mathscr{E}^t} \sum_{(v',u')\in\mathscr{E}^t} \max\{0, \gamma - S(v,u) + S(v',u')\}, \tag{12}$$

where v and u are suspicious hosts reported by the IPG detectors, and (v,u) are edges that may appear between them with a high probability of being malicious. For each potentially malicious edge (v,u), we choose a sample (v',u') as a normal edge that exists in $\mathscr{E}^t$ and is not reported as an anomaly. With this view, the minimization of the update loss function $\mathscr{L}^t_{\text{update}}$ pushes $S(v,u)$ to become larger and $S(v',u')$ to become smaller. With the enhancement from the IPG detectors, the IIG detection model can be efficiently updated to learn new attack patterns. The combination of the IPG and IIG can collectively detect APT activities from a comprehensive perspective.

7. Experimental Evaluation

The efficacy of the proposed method is explored in this section. We conducted all our experimental evaluations on Ubuntu 18.04 LTS, which possesses an Intel Xeon W-2133 3.60 GHz CPU, an NVIDIA GeForce RTX 2080 Ti GPU, and 64 GB RAM. The proposed graph embedding algorithms are implemented using Python's Deep Graph Library (DGL) that leverages the message passing mechanism to build GNNs. We evaluate our approach with different attack scenarios. For the Intrahost Provenance Graph and the corresponding IPG anomaly detector, we use the StreamSpot dataset to test the detection performance of our method at the system level. The Los Alamos National Lab (LANL) dataset is used to evaluate the IIG anomaly detector over the Interhost Interactive Graph from a network perspective.

7.1. Datasets

7.1.1. StreamSpot Dataset. The StreamSpot dataset is composed of 600 provenance graphs derived from 5 benign and 1 attack scenarios. The benign scenarios involve normal behaviors of playing video games, checking emails in Google Mail, browsing cnn.com, downloading files, and watching YouTube videos. The attack scenario involves malicious behaviors of a drive-by download attack triggered by browsing a malicious URL which exploits a Flash vulnerability and further gains root access to the victim host. For each scenario, 100 tasks were executed automatically on a Linux machine. The Linux SystemTap logging system is used to record system call traces from the start of a task until its termination on the machine. All system calls running on the machine (including the ones not from the task) are collected and used to construct provenance graphs for anomaly detection. The statistics of the dataset are summarized in Table 2.

7.1.2. LANL Dataset. The LANL dataset comprises 58 consecutive days of event data collected from the internal computer network of Los Alamos National Laboratory. It involves five data sources, including the network flow data collected from several key routers, DNS lookup events collected from internal DNS servers, process start and stop events collected from individual Windows computers and servers, authentication events collected from Windows-based individual computers and Active Directory servers, and a set of red team events that presents malicious behaviors deviating from normal computer activities. In total, the dataset comprises 1,648,275,307 events for 17,684 computers, involves 749 malicious events with 305 compromised computers which presents a typical APT campaign. We utilize three data sources to evaluate the IIG detector at the network level. Authentication events and internal network flow events are taken into account for detection of the second APT stage as attackers often attempt to compromise more hosts. To detect anomalous hosts that engage in communication with the C & C server and data exfiltration in the last phase of APT, attention is focused on the network flow and DNS lookup events from internal to external hosts.

7.2. Evaluation on the StreamSpot Dataset. We compare the IPG detector with three anomaly detection methods: StreamSpot [13], UNICORN [2], and Graph Attention Network (GAT) [22]. StreamSpot and UNICORN are state-of-the-art approaches for provenance-based APT anomaly detection using heuristic algorithms. GAT is a GNN algorithm that leverages self-attention layers to assign different weights to different neighbor nodes. We use 60% of the benign data to train the detection model and 40% of it along with all attack data for testing. The number of the metapath aggregated GNN layers is 3. The learning rate is 0.003. The weight decay for regularization is $5e-6$. As shown in Table 3, we employ precision, accuracy, recall, and F-score to evaluate different methods. The IPG detector performs best in all evaluation metrics compared with all baselines. In particular, the IPG detector has gained more than 25% improvement compared with the StreamSpot and more than 10% improvement compared with the UNICORN and GAT.

As shown in Table 4, we split the benign graphs into three datasets to fully explore the performance of our method. The Receiver Operator Characteristic (ROC) curves for all datasets are shown in Figure 6; note that the IPG detector is capable of ranking attack graphs correctly with AUC = 0.98 (area under the ROC curve). More specifically, on the ALL dataset, the IPG detector reaches an ideal performance with TPR = 0.98 and FPR = 0.1 when the threshold is set to 0.18 (which means that the graphs with reconstruction errors greater than 0.18 are classified as anomalies). Finally, we evaluate the influence of the training ratio on AUC and Average Precision (AP). Figure 7 shows the variations of AP and AUC with the training ratio varied from 10% to 90%. We note that our method achieves good detection performance when sufficient benign training graphs are employed.

7.3. Evaluation on the LANL Dataset. The LANL dataset is used to evaluate the performance of the IIG detector and the model update mechanism. The event types used for evaluation regarding authentication, network flow, and DNS lookup, which describe the network-level behaviors. The dataset properties are shown in Table 5, with all the three types of events are benign. We randomly select 600 malicious events from the red team file, accounting for 80% of the total. The 600 malicious operations involve 280 suspicious hosts and 254 suspicious users. The target environment involves 4,110 host nodes. We use 4,000 benign events (i.e., 2,000 authentication events, 1,000 flow events, and 1,000 DNS events) to examine FPRs, and the rest of benign events to train the IIG model. All attack data are used to examine the IIG detector's ability to identify malicious events.

Table 2: StreamSpot dataset summary.

Dataset	Scenario	Avg. $\|E\|$	Avg. $\|V\|$	# of graphs
StreamSpot	GMail	37,382	6,827	100
	Download	310,814	8,831	100
	CNN	294,903	8,990	100
	VGame	112,958	8,637	100
	YouTube	113,229	8,292	100
	Attack	28,423	8,891	100

To evaluate the performance of the IIG detector in different scenarios, we split the benign dataset into three subdatasets: authentication and DNS lookup events (AD), authentication and network flow events (AF), and all three types of benign events (AFD). Figure 8 presents the detection results in terms of ROC curves on the three datasets. Note that the IIG detector performs best with AUC = 0.90 when we use all benign events to train the detection model. The AUC of the model trained by the AD and AF datasets are 0.83 and 0.85, respectively. When we use the AFD for training and set the threshold to 0.27, the TPR of the IIG detector is 0.83, and the FPR does not exceed 0.05. It is evident from the ROC curve that increasing the number of event types for model training contributes to modeling network behaviors. As such, anomaly detection models incorporating multiple benign patterns are more likely to identify outliers.

We compare the IIG detector with four baseline methods (Tiresias [25], ensemble method [36], log2vec [3], and GAT [22]) to evaluate the effectiveness of our method. Tiresias is an advanced log-entry-level approach that leverages RNN to predict future events on a host. The ensemble method uses Principal Component Analysis, k-means clustering, and Median Absolute Deviation–based outlier detection to identify anomalies. Log2vec is a heterogeneous graph embedding-based approach that leverages random walk and word2vec to identify malicious events. We implement the IIG detector with a 3-layer edge feature-enhanced GNN. The learning rate is 0.002, and the weight decay for regularization is 5e-6.

As shown in Table 6, we employ precision, recall, *F*-score, and AUC to compare the performances of different methods. The LANL dataset is extremely unbalanced; thus, we do not use the accuracy as a metric, which cannot indicate performance of the detectors when anomalous data are scarce. The ROC curves of the IIG detector and the baselines are shown in Figure 9. Tiresias simply treats events with less than ten occurrences as anomalies. However, the rare events involve numerous benign patterns, which leads to poor performance with the precision and *F*-score less than 0.60. The ensemble method analyzes network events by identifying characteristic patterns as statistical features to detect APT behaviors, but the performance is not ideal. The performances of log2vec

Table 3: Detection results of different approaches on the StreamSpot dataset.

Experiment	Precision	Accuracy	Recall	*F*-score
StreamSpot	0.77	0.84	0.74	0.75
UNICORN	0.87	0.90	0.84	0.86
GAT	0.86	0.91	0.88	0.87
IPG detector	0.98	0.98	0.98	0.98

Table 4: StreamSpot subdatasets summary.

Dataset	Scenarios	Avg. $\|E\|$	Avg. $\|V\|$	# of graphs
ALL	GMail, download, CNN, VGame, YouTube	173,857	8,315	500
GVC	GMail, VGame, CNN	148,414	8,151	300
YDC	YouTube, download, CNN	239,648	8,705	300

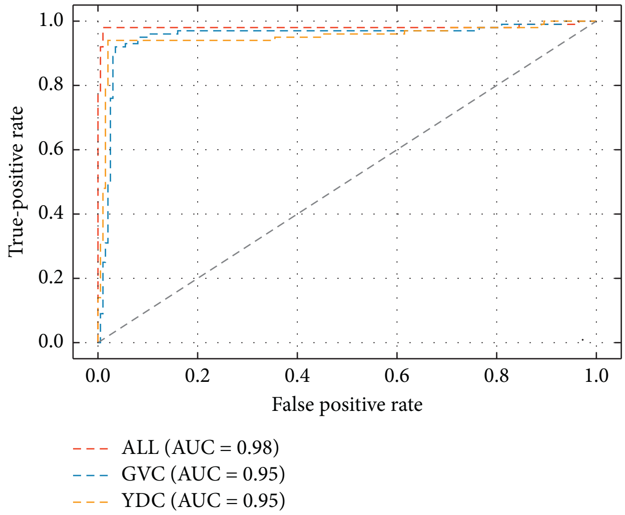


Figure 6: ROC curves of the IPG detector on the StreamSpot subdatasets.

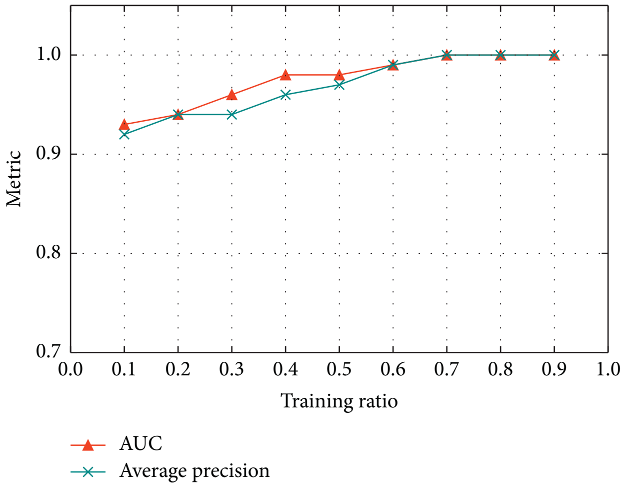


Figure 7: Performance of the IPG detector with different training ratios.

(AUC = 0.85) and GAT (AUC = 0.83) are similar. However, log2vec relies on a heuristic model that uses too many user-defined parameters and only uses statistical information from a fixed hop of neighbors, which ignores the global information of APT activities. GAT is unable to encode network-level events involving multidimensional edge features. The IIG detector incorporates edge features to perform graph embedding, thus producing significant improvement in all evaluation metrics (AUC = 0.9 and all other metrics are 0.83).

After executing malicious code and securing the entry point of the target environment, ATP attackers further search for critical assets and move laterally to exfiltrate information and undermine components. The malicious hosts reported from the system-level anomaly detectors can be further used to enhance the network-level anomaly detector. To evaluate the enhancement mechanism, we randomly label k malicious hosts assuming they are detected by the IIG detectors to update the trained IIG model. By increasing the number of reported hosts from 0 to 10, we investigate the impact of k in the IIG detector. The results are shown in Figure 10, where we employ AUC, AP, and TPR to verify the performance improvement. Compared with the IIG detector without enhancement, we can achieve improved performance as the number of reported malicious hosts increases. Specifically, when we label 10 malicious hosts, the IIG detector performs best with AUC = 0.95, TPR = 0.92, and AP = 0.91.

Table 5: LANL dataset summary.

Dataset	Event types	# of hosts	# of events	# of test events
LANL	Authentication	4,110	2,317,309	2,000
	Network flow	3,740	1,645,377	1,000
	DNS lookup	2,102	974,533	1,000
	Attack	280	600	600

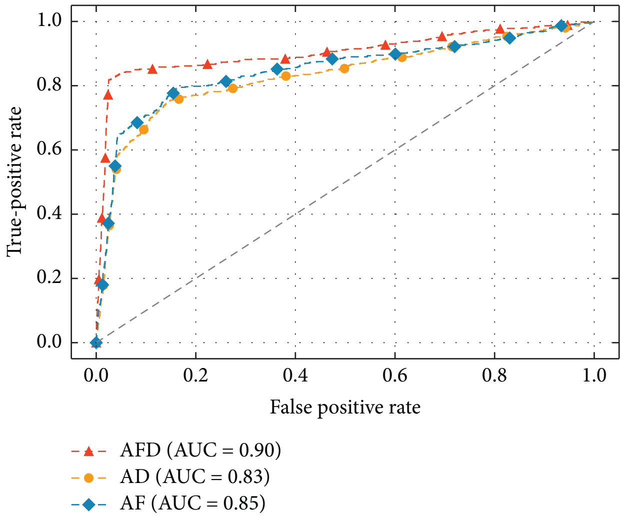


Figure 8: ROC curves of the IIG detector on the LANL subdatasets.

Table 6: Detection results of different methods on the LANL dataset.

Method	Precision	Recall	*F*-score	AUC
Tiresias	0.56	0.63	0.59	0.76
Ensemble method	0.62	0.68	0.65	0.80
Log2vec	0.72	0.78	0.74	0.85
GAT	0.69	0.73	0.71	0.83
IIG detector	0.83	0.83	0.83	0.90

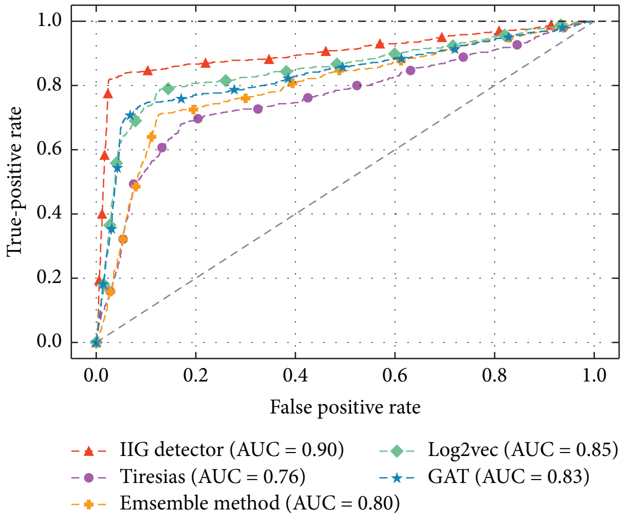


Figure 9: ROC curves of different methods on the LANL dataset.

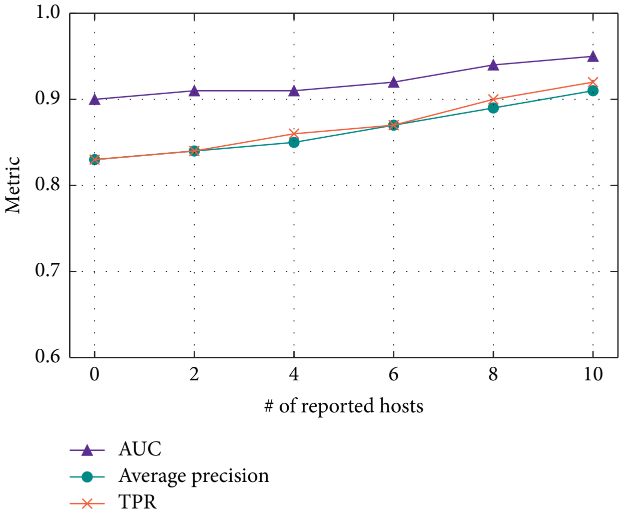


Figure 10: Performance of the IIG detector with different # of reported hosts.

8. Discussion

In this study, we propose a hierarchical framework with two detection models, IPG and IIG, to detect APT attacks. Currently, we evaluate the IPG detector on the StreamSpot dataset and the IIG detector on the LANL dataset separately. However, the two models need to be combined for a more adequate analysis. Besides, the identified anomalies should be further analyzed and mapped to the three-stage APT model. Thus, a complete dataset that records behaviors of a target environment at both the network and system levels is needed. In the future, we could implement a virtual environment to execute normal operations and simulate APT attacks. We could thus evaluate our approach adequately in such an environment.

9. Conclusion

To overcome the limitations of graph-based attack detection in APT studies, we propose a hierarchical approach for defending against APTs with novel attention-based GNNs. To comprehensively capture the behaviors of the full APT lifecycle, we propose a metapath aggregated GNN and an edge feature-enhanced GNN to identify anomalies at both the system and network levels. Besides, we present an enhancement mechanism to dynamically update the IIG model with reported hosts from the IPG detectors in the hierarchical detection framework. The evaluations show that our methods outperform the state-of-the-art baselines.

Data Availability

The StreamSpot data and LANL data used to support the results of this work are publically available at https://github.com/sbustreamspot/sbustreamspot-data/ and https://csr.lanl.gov/data/cyber1/.

Conflicts of Interest

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

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China, under Grant 2019YFB2102002, and in part by the Key Research and Development Program of Jiangsu Province, under Grant BE2019012.

References

[1] S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar, and V. N. Venkatakrishnan, "Holmes: real-time APT detection through correlation of suspicious information flows," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1137–1152, IEEE, San Francisco, CA, USA, May 2019.

[2] X. Y. Han, T. Pasquier, A. Bates, J. Mickens, and M. Seltzer, "UNICORN: runtime provenance-based detector for advanced persistent threats," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2020.

[3] F. Liu, Y. Wen, D. X. Zhang, X. H. Jiang, X. Y. Xing, and D. Meng, "Log2vec: a heterogeneous graph embedding based approach for detecting cyber threats within enterprise," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1777–1794, ACM, London, UK, November 2019.

[4] A. Zimba, H. Chen, Z. Wang, and M. Chishimba, "Modeling and detection of the multi-stages of advanced persistent threats attacks based on semi-supervised learning and complex networks characteristics," *Future Generation Computer Systems*, vol. 106, pp. 501–517, 2020.

[5] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A survey on advanced persistent threats: techniques, solutions, challenges, and research opportunities," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1851–1877, 2019.

[6] X. Yan and J. Zhang, "Early detection of cyber security threats using structured behavior modeling," *Transactions on Information and System Security*, vol. 5, pp. 1–19, 2013.

[7] A. S. K. Pathan, *The State of the Art in Intrusion Prevention and Detection*, Auerbach Publications, Boca Raton, FL, USA, 2014.

[8] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the SIGSAC Conference on Computer and Communications Security*, pp. 1285–1298, ACM, Dallas, TX, USA, November 2017.

[9] P. Parveen, N. McDaniel, V. S. Hariharan, B. Thuraisingham, and L. Khan, "Unsupervised ensemble based learning for insider threat detection," in *Proceedings of the 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pp. 718–727, Amsterdam, Netherlands, September 2012.

[10] M. Du, Z. Chen, C. Liu, R. Oak, and D. Song, "Lifelong anomaly detection through unlearning," in *Proceedings of the SIGSAC Conference on Computer and Communications Security*, pp. 1283–1297, ACM, London, UK, November 2019.

[11] B. Debnath, M. Solaimani, M. A. G. Gulzar, N. Arora, C. Lumezanu et al., "LogLens: a real-time log analysis system,," in *Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1052–1062, IEEE, Vienna, Austria, July 2018.

[12] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[13] E. Manzoor, S. M. Milajerdi, and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1035–1044, San Francisco, CA, USA, August 2016.

[14] Z. Li, Q. A. Chen, R. Yang, and Y. Chen, "Threat detection and investigation with system-level provenance graphs: a survey," 2020, https://arxiv.org/abs/2006.01722.

[15] M. N. Hossain, S. M. Milajerdi, J. Wang et al., "Real-time attack scenario reconstruction from COTS audit data," in *Proceedings of the 26th USENIX Conference on Security Symposium*, pp. 487–504, Vancouver, Canada, August 2017.

[16] S. M. Milajerdi, B. Eshete, R. Gjomemo, and V. Venkatakrishnan, "Poirot: aligning attack behavior with kernel audit records for cyber threat hunting," in *Proceedings of the SIGSAC Conference on Computer and Communications Security*, pp. 1813–1830, ACM, London, UK, November 2019.

[17] K. Pei, Z. S. Gu, B. Saltaformaggio et al., "Hercule: attack story reconstruction via community discovery on correlated log graph," in *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pp. 583–595, ACM, Los Angeles, CA, USA, December 2016.

[18] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*, pp. 701–710, New York, NY, USA, August 2014.

[19] A. Grover and J. Leskovec, "Node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*, pp. 855–864, San Francisco, CA, USA, August 2016.

[20] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graph," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, Curran Associates Inc, Long Beach, CA, USA, December 2017.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the International Conference on Learning Representations*, Toulon, France, April 2017.

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, April 2018.

[23] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep anomaly detection on attributed networks," in *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 594–602, SIAM, Calgary, Canada, May 2019.

[24] L. Zheng, Z. P. Li, J. Li, Z. Li, and J. Gao, "AddGraph: anomaly detection in dynamic graph using attention-based temporal GCN," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence Main Track*, pp. 4419–4425, Macao, China, August 2019.

[25] Y. Shen, E. Mariconti, P. A. Vervier, and G. Stringhini, "Tiresias: predicting security events through deep learning,"

in *Proceedings of the SIGSAC Conference on Computer and Communications Security*, pp. 592–605, ACM, Toronto Canada, October 2018.

[26] A. Gehani and D. Tariq, "SPADE: support for provenance auditing in distributed environments," in *Proceedings of the 13th International Middleware Conference*, pp. 101–120, Montreal, Canada, December 2012.

[27] T. Pasquier, X. Han, M. Goldstein et al., "Practical whole-system provenance capture," in *Proceedings of the 2017 Symposium on Cloud Computing*, pp. 405–418, ACM, Santa Clara, CA, USA, September 2017.

[28] S. T. King and P. M. Chen, "Backtracking intrusions," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 223–236, 2003.

[29] Y. Liu, M. Zhang, D. Li et al., "Towards a timely causality analysis for enterprise security," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2018.

[30] W. U. Hassan, S. Guo, D. Li et al., "Combatting threat alert fatigue with automated provenance triage," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2019.

[31] P. Moriano, J. Pendleton, S. Rich, and L. J. Camp, "Insider threat event detection in user-system interactions," in *Proceedings of the 2017 International Workshop on Managing Insider Security Threats*, pp. 1–12, ACM, Dallas, TX, USA, October 2017.

[32] Y. Han, A. P. Li, and R. Jiang, "Needle in a haystack: attack detection from large-scale system audit," in *Proceedings of the 2019 IEEE 19th International Conference on Communication Technology (ICCT)*, pp. 1418–1426, Xi'an, China, October 2019.

[33] E. Hutchins, M. Cloppert, and R. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," in *Proceedings of the 6th International Conference on Information Warfare and Security*, Washington, DC, USA, March 2011.

[34] D. J. Pohly, S. McLaughlin, P. McDaniel, and K. Butler, "Hi-fi: collecting high-fidelity whole-system provenance," in *Proceedings of the Computer Security Applications Conference*, pp. 259–268, Orlando, FL, USA, December 2012.

[35] A. M. Bates, D. Tian, K. R. Butler, and T. Moyer, "Trustworthy whole-system provenance for the Linux kernel," in *Proceedings of the 24th USENIX Security Symposium*, pp. 319–334, USENIX, Washington, DC, USA, August 2015.

[36] A. Bohara, M. A. Noureddine, A. Fawaz, and W. H. Sanders, "An unsupervised multi-detector approach for identifying malicious lateral movement," in *Proceedings of the 2017 IEEE 36th Symposium on Reliable Distributed Systems*, pp. 224–233, IEEE, Hong Kong, China, September 2017.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9919030, 16 pages
https://doi.org/10.1155/2021/9919030

Research Article

An Adaptive Communication-Efficient Federated Learning to Resist Gradient-Based Reconstruction Attacks

Yanbin Li, Yue Li, Huanliang Xu, and Shougang Ren

College of Artificial Intelligence, Nanjing Agricultural University, Nanjing 210095, China

Correspondence should be addressed to Shougang Ren; rensg@njau.edu.cn

Received 10 March 2021; Revised 2 April 2021; Accepted 13 April 2021; Published 22 April 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Yanbin Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The widely deployed devices in Internet of Things (IoT) have opened up a large amount of IoT data. Recently, federated learning emerges as a promising solution aiming to protect user privacy on IoT devices by training a globally shared model. However, the devices in the complex IoT environments pose great challenge to federate learning, which is vulnerable to gradient-based reconstruction attacks. In this paper, we discuss the relationships between the security of federated learning model and optimization technologies of decreasing communication overhead comprehensively. To promote the efficiency and security, we propose a defence strategy of federated learning which is suitable to resource-constrained IoT devices. The adaptive communication strategy is to adjust the frequency and parameter compression by analysing the training loss to ensure the security of the model. The experiments show the efficiency of our proposed method to decrease communication overhead, while preventing privacy data leakage.

1. Introduction

In recent years, Internet of Things (IoT) has had great popularity in different aspects of modern life and a huge amount of IoT services are emerging. In the IoT area, user devices generate a large amount of data that can be used to improve the user experience of the intelligence system. However, the extensive users' data processing from the IoT device brings some privacy problems [1]. As the IoT devices can be deeply involved in users' private data, the data generated by them will contain privacy-sensitive information [2–4]. To tackle the privacy challenges and encourage clients to proactively participate in IoT services, federated learning enables training a deep learning model across different participants in a collaborative manner. It provides the privacy of clients to keep their original data training on their own devices, while jointly learn a global model by sharing only local parameters with the server.

However, several recent works have shown that the privacy in federated learning is insufficient for protecting the local training data from gradient-based reconstruction attacks [5–7]. The wide malicious devices in IoT make it vulnerable to these type attacks based on shared parameters. The first type of attack is GAN-based attacks. Hitaj and Perez-Cruz proposed a GAN-based attack against collaborative deep learning on a malicious client, which infers sensitive information from another client successfully [8]. Based on this work, an improve GAN with a multitask discriminator was proposed to enable a malicious server to discriminate category, reality, and client identity of input samples simultaneously [9]. Another type of gradient-based reconstruction attacks is Deep Leakage from Gradients (DLG), which was proposed by Zhu et al. to reveal the training data from gradients [6]. The main idea of DLG is to generate dummy data and labels via matching the dummy gradients to the shared gradients. It has been used in many following works to perform the privacy leakage attacks on federated learning [5, 7]. The GAN-based attack uses GAN to generate pictures that look similar to the training images, while DLG aims at revealing the complete training data from gradients. These two types of attacks both utilize gradient to reconstruction.

To guarantee the privacy of federated learning, there are many privacy techniques for preventing indirect leakage.

Cheng et al. presented a FL-EM-GMM algorithm to make model training without data exchange for protecting privacy [10]. Secure multiparty computation (SMC) involves multiple parties and provides security proof to guarantee complete zero knowledge so that each party knows nothing except its own input and output. It has been used for model training and verification without users revealing sensitive data [11–13]. However, the secure aggregation requires gradients to be integers, which makes it is not compatible with most CNNs. A general method named differential privacy involves adding noise to the training data or obscuring certain sensitive attributes so that the third party could not distinguish the individual information [14–16]. This method usually decreases the accuracy. However, theGAN-based attack is resist against to a certain-level differential privacy [8]. Asad et al. proposed the FedOpt algorithm using homomorphic encryption to protect the privacy of users [17]. But homomorphic encryption increased the model upload time and the system burden, which may increase the system overhead in a bandwidth-limited server system. In addition, they ignore the relationship between efficiency and privacy.

To overcome the performance bottleneck is to apply optimization technologies in federated learning. There are numerous variants of gradient quantization or sparsification and communication delay have been proposed to different distributed deep learning tasks to reduce the communication cost. Han et al. proposed a fairness-aware gradient sparsification method to minimize the overall training time [18]. Zhou et al. proposed a privacy-preserving multidimensional data aggregation scheme, which has great advantages in the communication overhead [19]. In addition, the adaptive communication strategy was adopted to save communication delay and improve convergence speed [20]. There are few works that discuss the impact of these optimization technologies on the security under gradient-based reconstruction attacks. The experiments showed that gradient compression and sparsification could mitigate the leakage of DLG [6, 21].

In this paper, we discuss the security of the federated learning model with different optimization technologies comprehensively. Based on our analysis, the optimization technologies used to reduce communication cost may also improve the resistance against gradient-based reconstruction attacks. To promote the efficiency and security simultaneously, we propose a defence strategy of federated learning without extra high overhead countermeasures, which are not suitable to resource-constrained IoT devices. Our strategy aims at reducing the communication overhead in IoT environment and achieving higher security against gradient-based reconstruction attacks. The experiments on the open source dataset have shown that our method achieves a relative low training loss and prevents from gradient-based reconstruction attacks.

The remainder of the paper is organized as follows. Section 2 describes the optimization technologies to improve the efficiency of FL and the two type gradient-based reconstruction attacks for FL. In Section 3, we discuss the relationship between the optimization and security. Based on the results, we introduce a new method of FL to improve the efficiency and security simultaneously in Section 4. The experimental results are shown in Section 5. Finally, we provide the conclusion.

2. Background

2.1. Efficiency Optimization of Federated Learning. The surge of massive data has led to significant interest in distributed algorithms for scaling computations in the context of machine learning and optimization. The baseline communication protocol is used in many early federated learning implementations: the client sends a full vector of local training parameter update back to the federated learning server in each round. In this context, the current research is focused on how to reduce the transfer cost of model parameters to make it more efficient in terms of communication, of which the gradient compression and periodic methods are intensively researched.

2.1.1. Gradient Compression. Gradient quantization or sparsification is used to reduce the communication cost through gradient compression. Strom proposed a compression and quantization-based approach to compress single communications and introduced the concept of gradient residuals [22]. Firstly, the participating node k computes the local gradient ΔW_k by adding the local gradient ΔW_k to the previously residual gradient residual ΔW_k^r. If the new gradient is larger than the threshold T, the index and threshold T of that gradient are encapsulated in message M and the gradient residuals are updated: $\Delta W_k^r = \Delta W_k^r - T$. If the new negative gradient is less than threshold $-T$, the index and threshold $-T$ of the gradient are also encapsulated in message M, and the gradient residuals are updated: $\Delta W_k^r = \Delta W_k^r + T$. Finally, the compression message M is sent to other nodes.

Given that the exact selection of the threshold T is difficult in practice, Aji and Heafield proposed a heuristic method for threshold compression [23]. The unique feature of this method is the dynamic selection of thresholds, which reduces the difficulty of threshold selection by setting a discard rate R, sorting the sampled gradient values by absolute values, and taking the number with the $(1 - R\%)$ largest absolute value of the gradient as the current threshold. Tian et al. proposed a novel sketch-based framework (DiffSketch) for distributed learning [24]. The framework can protect privacy using federated learning and compressing the parameters. But some existing attacks can already steal privacy information in federated learning, and only compressing the parameters could not guarantee the data privacy. We would like to seek a balance between compression and communication frequency that protects privacy and ensures accuracy.

2.1.2. Periodic. The communication delay approach is another solution to the above bottleneck, which differs from the gradient sparse and quantitative optimization perspective. The former significantly reduces the number of communication rounds by increasing the local computational cost appropriately, while the latter is to reduce the cost of communication per round. The two can be complementary, but not contradictory.

Though period-average gradient descent can significantly reduce the number of communication rounds through delayed communication, it also increases the local computational cost, and the appropriate communication frequency is not easy to select. High frequent communication leads to huge communication rounds, but eventually it converges to a smaller loss, while sparse communication reduces the cost of communication, but the results of the federated learning model is worse. Therefore, to solve the above issues, Wang proposed an adaptive communication strategy ADACOMM, which divided the training phase into subphases and tried to find the optimal communication frequency for each phase [20]. Before the start of the new phase, ADACOMM was used to select the frequency by the training loss.

The communication frequency update rule is shown in the following equation:

$$\tau_l = \lceil \sqrt{\frac{F\left(X_{t=lT_0}\right)}{F\left(X_{t=0}\right)}\tau_0} \rceil. \tag{1}$$

As the training proceeds, the loss $F(X_{t=lT_0})$ becomes smaller and the frequency τ_l becomes smaller, i.e., the local computation round becomes smaller and the communication frequency increases gradually. Finally, the model will converge with fewer iterative rounds according to the ADACOMM.

2.2. Gradient-Based Reconstruction Attacks. The original idea of the federated learning was to build global models based on the gradient parameters that are distributed across multiple devices and to prevent data leakage. Potential loopholes are found in some research in the gradients shared by federated learning, which can be divided into two main categories: GAN-based attacks and DLG attacks. The procedure of the two types of attacks is depicted in Figure 1.

2.2.1. GAN-Based Attacks. GAN is proposed to implement a novel class of active inference attacks on deep neural networks in a collaborative setting. Specifically, the generator G attempts to imitate the data from target distribution to make the "fake" data indistinguishable to the adversarial supervisor D. There may be a setup defensible to attacks, which may be achieved by setting stronger privacy guarantees, releasing fewer parameters, or establishing tighter thresholds. However, as proved by the results in this article, such measures lead to models that are unable to learn or worse performance than models trained on centralized data. Therefore, we consider solving the problem from a combination of approaches.

2.2.2. DLG Attacks. Zhu et al. presented an approach which shows the possibility to obtain private training data from the publicly shared gradients [6]. In their Deep Leakage from Gradient (DLG) method, they synthesized the dummy data and corresponding labels with the supervision of shared gradients. Specifically, they start with random initialization of pseudodata and labels. Virtual gradients are computed on the current shared model in the distributed setup. By minimizing the difference between the virtual gradient and the shared real gradient, they iteratively update the virtual data and labels simultaneously. Although DLG works, we find that it could be affected by a number of factors that affect the quality of the images generated by federated learning efficiency.

2.3. Notation. In order to express with conciseness and standardization, we stipulate the letters' notation of some indicators and show the main hyperparameter settings and notations in Table 1. CE is a communication compression ratio index, which is also one of the most important indexes for evaluating communication efficient algorithms. E_0 is the fixed rounds of updating. $\text{avg}_{\text{parameter}}$ is the mean parameters of all iteration communication. acc_{90} is the accuracy reached 90% for the first time. $\max_{\text{acc}}$ is the maximum of the accuracy in all iteration.

3. The Relationship between Communication Efficiency and Security

Recent improvements have been focused on communication cost in federated learning. The main approaches are to reduce the communication overhead and improve the overall efficiency of federated learning. The goal can be achieved by reducing the communication frequency and compressing the parameters. This section introduces the evaluation indexes to measure the security threats to federated learning; based on this, we discuss the relationship between communication optimization methods and security under the gradient-based reconstruction attacks in federated learning.

3.1. Evaluation Metrics. Privacy threats in federated learning are mainly recovery training dataset images and image pixels that imply private information about the user, so the image similarity metric can be referred as a security evaluation metric.

Attack success rate (ASR) refers to the percentage of successful attackers recovering the local training data victim. The metrics for determining the success of the attack are different for various attack strategies. In the GAN-based attacks, the accuracy of the recovered image label category shall prevail; however, in the DLG attacks, the similarity between the reconstructed image and the original image can be used as a criterion for success. The attack success rate (ASR) is the percentage of successfully reconstructed training data to the number of attacked training data.

An iterative attack is a situation in which a malicious attacker recovers the original data attacked through multiple iterative rounds. The criteria for determining the success of an attack is the same as ASR.

Structural Similarity (SSIM) is usually used as an index to measure the similarity of two images. SSIM is based on the

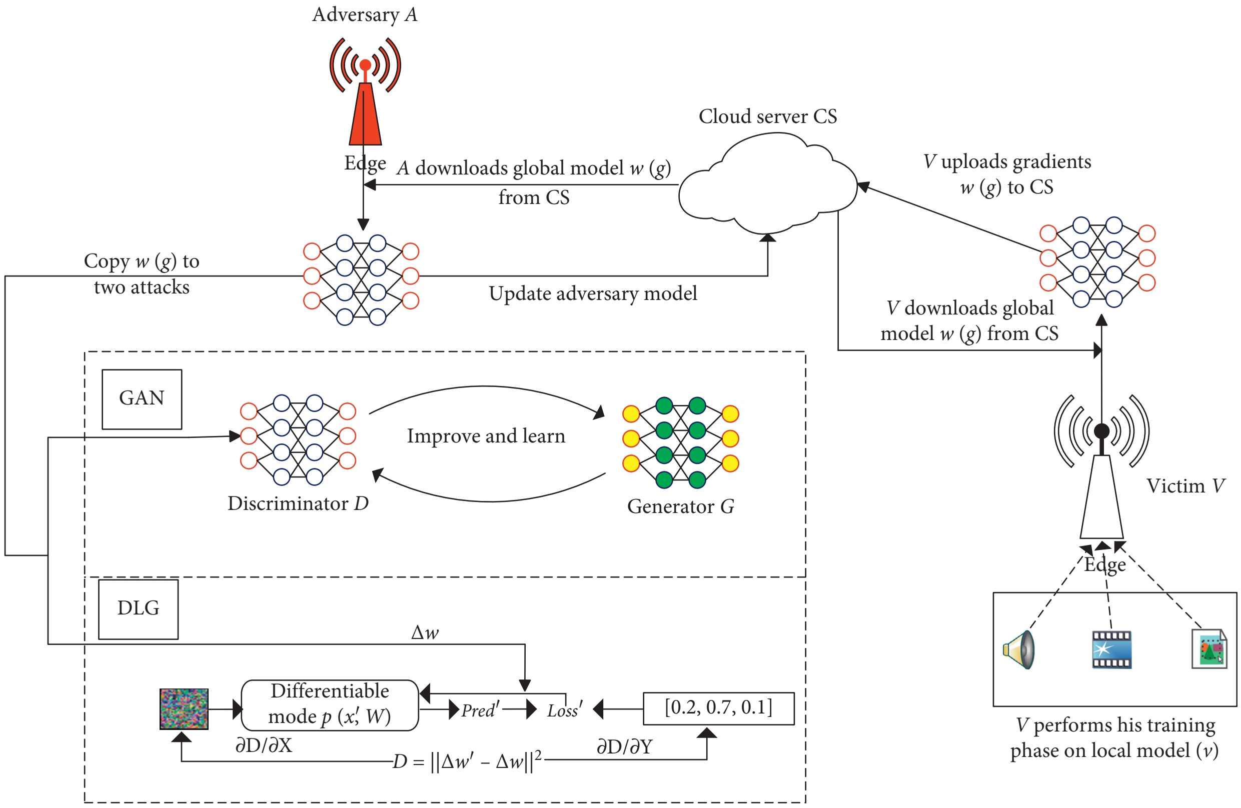

Adversary A
Edge
A downloads global model w (g) from CS
Cloud server CS
V uploads gradients w (g) to CS
Copy w (g) to two attacks
Update adversary model
V downloads global model w (g) from CS
GAN
Improve and learn
Discriminator D
Generator G
Victim V
Edge
DLG
Δw
Differentiable mode p (x′, W)
Pred′
Loss′
[0.2, 0.7, 0.1]
∂D/∂X
D = ||Δw′ − Δw||²
∂D/∂Y
V performs his training phase on local model (v)

Figure 1: The gradient-based reconstruction attacks on federated learning in IoT.

Table 1: Hyperparameter settings and letter representation.

Notation	Denote
M	The number of all clients
C	The number of compression
f	The number of frequency
CE	Compression rate of single communication
E_0	The fixed rounds of updating
$\text{avg}_{\text{parameter}}$	The mean parameters of all iterations' communication
acc_{90}	The number of iterations when test accuracy is beyond 90 for the first time in all iterations
$\max_{\text{acc}}$	The maximum of the accuracy in all iterations

perception model to measure the structural similarity between two images. Due to the outstanding performance of this indicator, it has been widely used in fields such as measuring video quality and image deblurring. Given two images x and y, then SSIM can be expressed as

$$\text{SSIM}(x, y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{2}$$

where μ_x and μ_y are estimated as the mean intensity, and the luminance comparison function is then a function of μ_x and μ_y. σ_x and σ_y are the unbiased estimate in the discrete form, and the comparison of the two signal is used as the contrast comparison. The constant c_1 and c_2 are to avoid instability when other signals are close to zero.

MSE is a signal fidelity measure. MSE refers to the root mean square deviation. The mean square error function is used to measure the similarity between the attacker's reconstructed image y and its true value x. Usually, it is assumed that one of the signals is a pristine original, while the other is distorted or contaminated by errors. The data recovered by the attacker is more similar to the real data with smaller MSE. The following formula is usually used to calculate MSE:

$$\text{MSE}(x, y) = \frac{1}{M}\sum_{i=1}^{M}\left(x(i) - y(i)\right)^2, \tag{3}$$

where x and y are two finite-length discrete signals (e.g., visual images), where M is the number of signal samples (pixels, if the signals are images) and $x(i)$ and $y(i)$ denote the values of the ith samples in x and y, respectively.

3.2. Relationships between Efficiency and Security. Hitaj et al. proposed the GAN-based attacks, to which the impact of optimization methods has not been researched. Therefore, this section is to study the effect of communication performance factors on the security of the two types of attacks [8]. We perform the experiments on the MNIST and AIFAR-100, which were used as the validation datasets in DLG work [6]. To defense the DLG attacks, the author experimented to defend by gradient compression. We reproduce some experiments according to the source code given by the authors, and the results show that parametric compression of the recovered images has obvious artifact pixels at 10% compression. This result is better than that described in that article. Firstly, the effect of a change in communication frequency on the security of federated learning is shown under the two types of attacks. Secondly, we discuss the effect of the two types of attacks under different parameter compression rates. The two evaluative metrics, SSIM and MSE, are used to determine the results of the attacks. Finally, we summarize the defensive effects of the two factors affecting the efficiency of federated learning on its security.

3.2.1. Relationships between Frequency and Security. Since the DLG method is a pixel-level reconstruction, the number of categories in the original datasets has no effect on it. The successful attacks of the DLG attacks are influenced by the pixel size of the original image. The GAN is very different. The method based on GAN-based attacks is label-level image reconstruction. The number of categories in the original data set determines the classification effect of the classifier, which in turn affects the classification effect of the discriminator, and ultimately affects the generator generation image quality. Therefore, the DLG attacks' method can achieve better attack effects on both the MNIST and CIFAR datasets, while the GAN-based attacks' method performs worse in the CIFAR100 dataset.

Figure 2 is the experimental result when the number of DLG attacks' iteration rounds is set to 500, and the communication frequency is 1. The image is reconstructed by printing the attack every 10 rounds. It can be seen from the figure that, after about 60 iterations, the original image can be considered successfully attacked.

Figure 3 shows the 36 three-category reconstructed images generated by the GAN method after 500 rounds of attack. The generated image can be clearly recognized as 3 by the human eyes, so this attack can be considered effective. The GAN method can generate false images in batches, which is very efficient in scenarios where the image quality is not high and only the category requirements are required.

Reducing the communication cost is one of the optimization goals of federated learning. The method to change the local communication frequency can alleviate the bottleneck problem caused by communication effectively. On the contrary, the change of communication frequency also caused a change in the security of federated learning. In this series of experiments, we explore the relationship between communication frequency and DLG attacks. The experiment includes 15 groups, with the communication frequency set to different values from 1 to 50 and the learning rate of 0.001. We count two indicators, SSIM and MSE, and visualize the reconstructed image after the attacks. The specific experimental results are shown in Table 2.

From the statistical data in the table, we can clearly see that, as the number of local training rounds increases (the communication frequency decreases), the similarity between the image generated by the statistical reconstruction and the original image becomes smaller and smaller, showing an opposite linear relationship. The experimental results show that, within a certain limit, reducing the communication frequency cannot only reduce the communication cost of federated training but also increase the difficulty of the DLG attacks against other client data attacks, which improves the security of the federated learning system.

Figure 4 shows the initial messy image, the original image, and 15 groups of attacks' reconstruction images under different communication frequencies. The visualization results are consistent with Table 2, and the image quality recovered by the attacker is getting worse. Similarly, we count the experimental results of this method on the MNIST dataset, and the above experimental phenomenon can also be found.

We also count the experimental results of the GAN-based attacks on the MNIST dataset. Since this method is more sensitive to the communication frequency, the experiment only sets 5 different frequency values. From the experimental results in Table 3, we can find that the SSIM value is smaller and the MSE value is larger. That is, the image quality reconstructed by the GAN-based attacks' method is average, but the category information is still there, so the applicability of the two indicators of image similarity becomes weaker here.

Figure 5 is the reconstructed images corresponding to the above settings. The image is tending to get blurred, and its category information is gradually lost. The experimental results are consistent with the experimental results of DLG.

From the above experiments, we can find that changing the communication frequency is one of the key factors affecting the attacker's success in the federated learning environment. We can find that the greater the number of local training cycles, the more difficult the attacks, that is, the more secure the client data during the training of the federated learning system.

3.2.2. Relationships between Compression and Security. Parameter compression (gradient sparseness) is often used in federated learning algorithms to reduce the amount of communication between the client and the parameter server, thereby to improve training efficiency. Previous studies have pointed out that this strategy will also affect the difficulty for potential attackers to recover other client data. In order to further explore the potential relationship between federated learning performance and security, we also set up multiple sets of comparative experiments and make statistics on relevant indicators and visualized reconstructed images.

Table 4 is the statistics of comparative experiments conducted under the same communication frequency and different communication compression ratios. We control different communication compression ratios by setting different thresholds. During the communication process, the client only passes the parameters (gradients) that exceed the

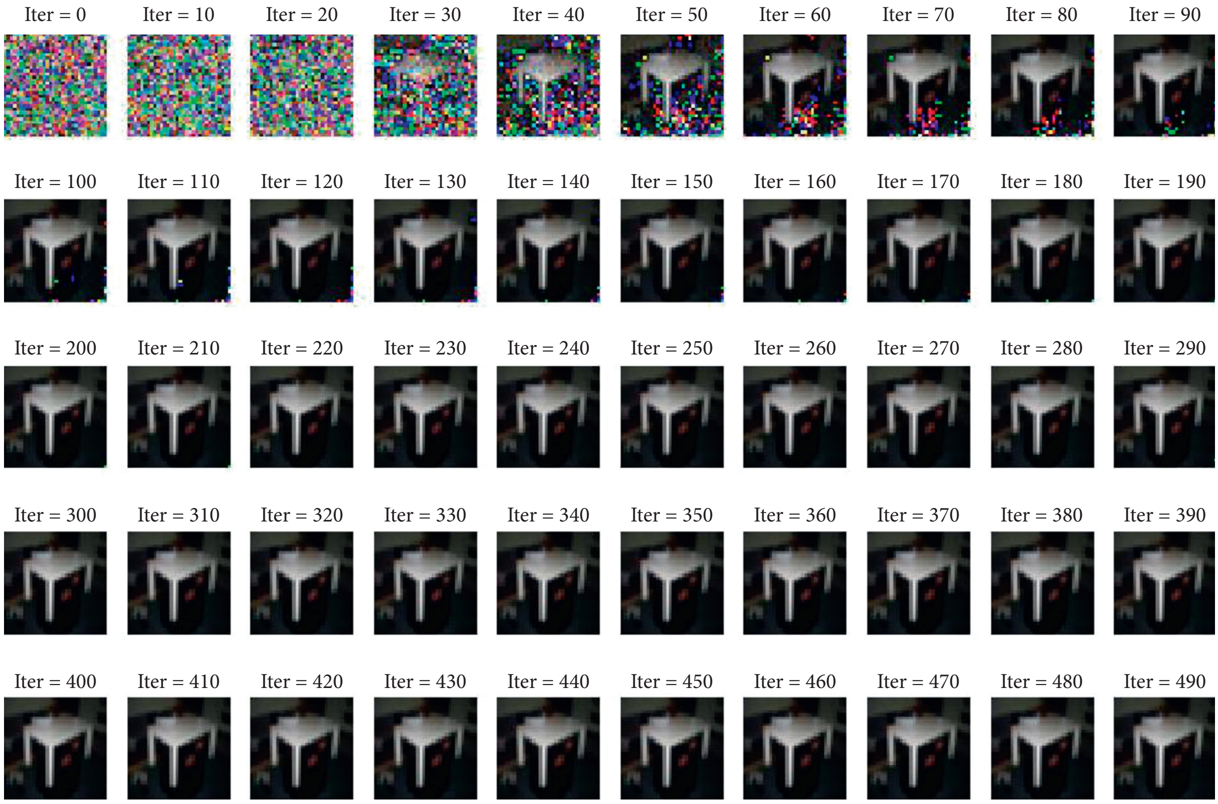


Figure 2: The results of DLG attacks on images from CIFAR-100.

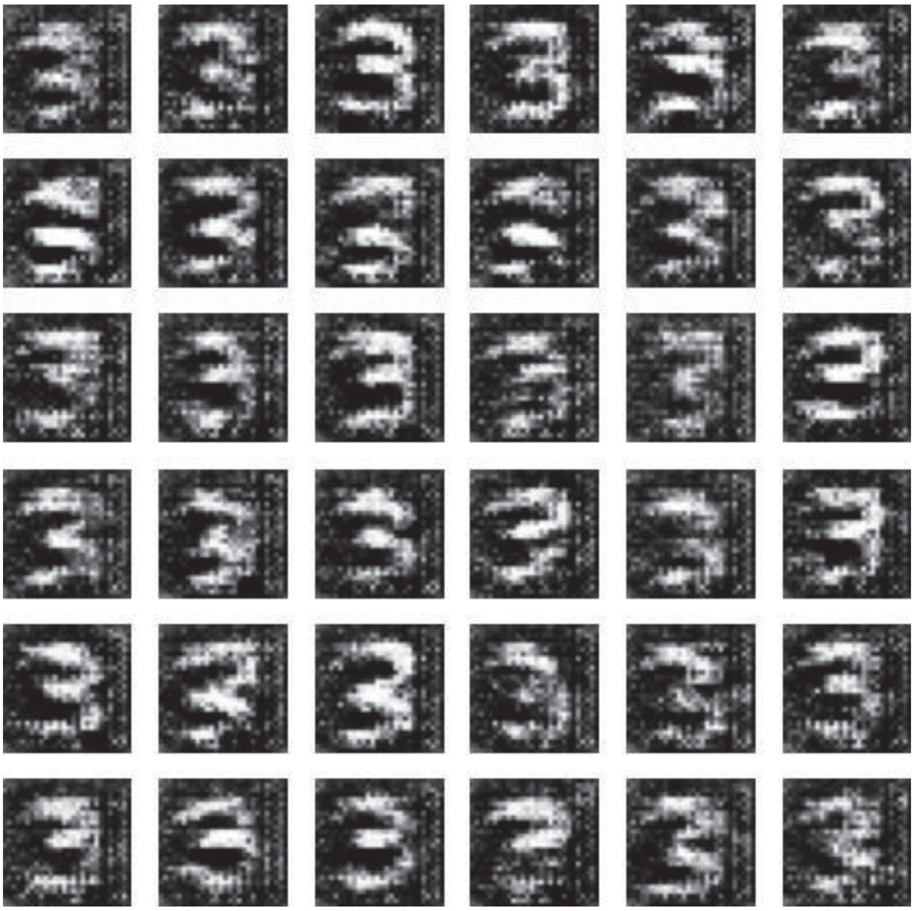


Figure 3: The reconstructed images generated by the GAN-based attacks after 500 rounds.

Table 2: The evaluation metrics of the DLG attacks with the CIFAR100 dataset in different frequencies.

Method	Iterations	Communication frequency	SSIM	MSE
DLG	500	1	0.9985	0.3300
DLG	500	2	0.9961	0.5437
DLG	500	3	0.9892	1.4399
DLG	500	4	0.9815	2.7646
DLG	500	5	0.9731	4.3394
DLG	500	6	0.9631	6.3177
DLG	500	7	0.9521	8.3981
DLG	500	8	0.9427	10.8416
DLG	500	9	0.9325	12.7527
DLG	500	10	0.9215	15.2681
DLG	500	15	0.8824	23.8498
DLG	500	20	0.8427	30.3106
DLG	500	30	0.7856	36.5652
DLG	500	40	0.7440	40.5796
DLG	500	50	0.7170	42.4485

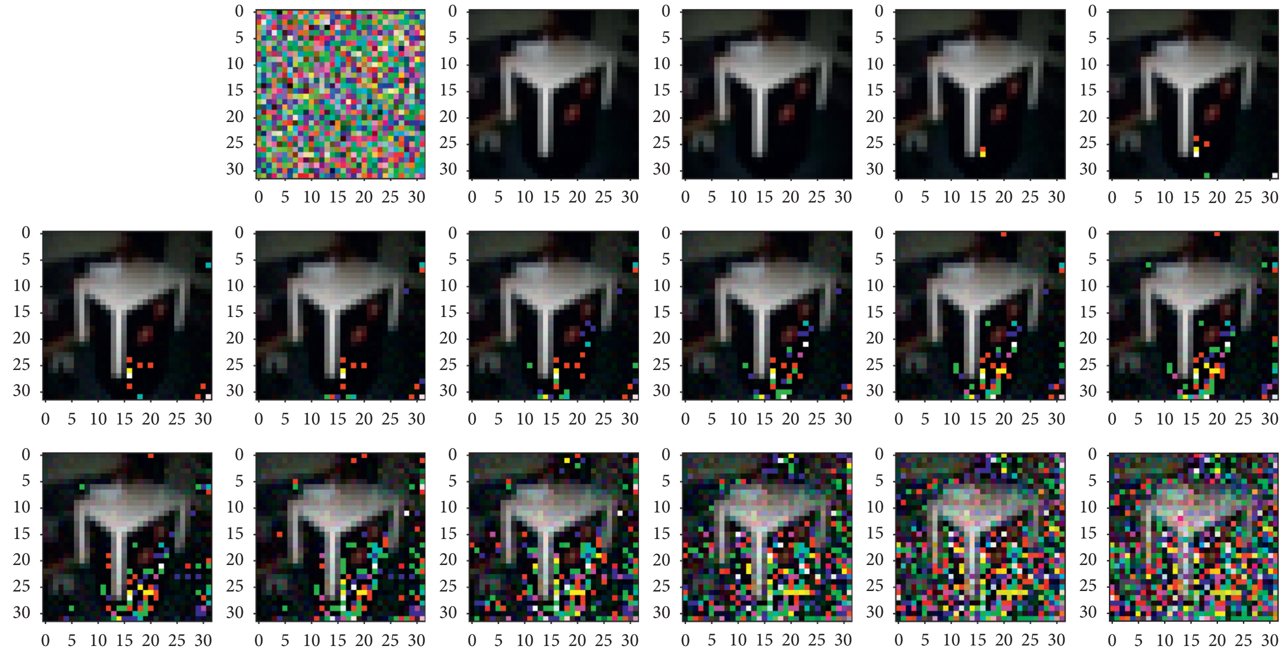


Figure 4: The results of the DLG attacks with the CIFAR100 dataset in different frequencies.

Table 3: The evaluation metrics of the GAN-based attacks with the MNIST dataset in different frequencies.

Method	Iterations	Communication frequency	SSIM	MSE
GAN	100	1	0.6289	41.9540
GAN	100	2	0.6347	40.3287
GAN	100	3	0.6110	43.9704
GAN	100	4	0.6009	42.4333
GAN	100	5	0.5914	46.7876

threshold to the parameter server for aggregation. The experiment set 6 thresholds of different levels, and the learning rate was uniformly set to 0.001. After 500 rounds of attack iterations, the image similarity index was counted, and the attacks' results were visualized.

Figure 6 shows that parameter compression plays a role in suppressing the GAN-based attacks' mode as well. Since GAN-based attacks perform more frequent interactions for federated learning, the effect of the image is no longer evident when the parameters are compressed to 90. Although the applicability of the two security metrics is weak, the overall trend in Table 5, and the reconstructed image in Figure 7 can reflect the progressively worsening effect of the attack.

It can be seen from the above experimental results that proper parameter compression can effectively avoid the leakage of local data and also reduce the single communication cost. However, excessive compression will adversely affect the training of the global model. When changing the degree of sparsity, we can see that the attacks still cannot be successful when the compression rate reaches 90%. So we can achieve a balance between compression and security by setting appropriate parameter compression thresholds.

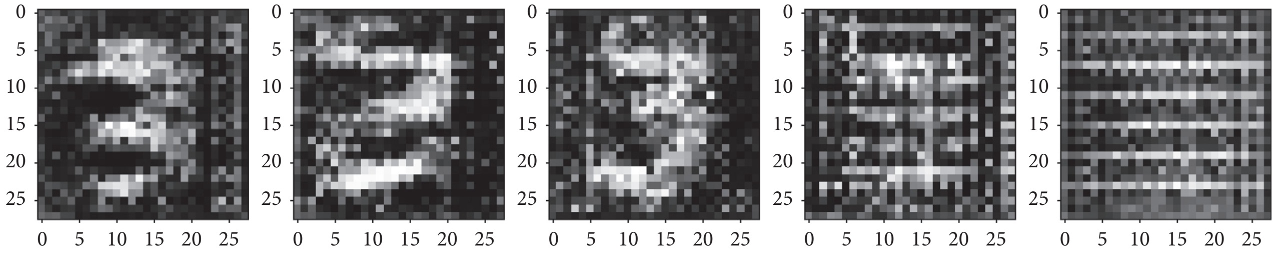


Figure 5: The results of the GAN-based attacks with the MNIST dataset in different frequencies.

Table 4: The evaluation metrics of the DLG attacks with the CIFAR100 dataset in different compression.

Iterations	CE	SSIM	MSE
500	1	0.9987	0.2948
500	0.98	0.9825	2.7182
500	0.96	0.9158	17.5191
500	0.94	0.8589	28.4093
500	0.92	0.7963	37.3693
500	0.90	0.6201	46.2852

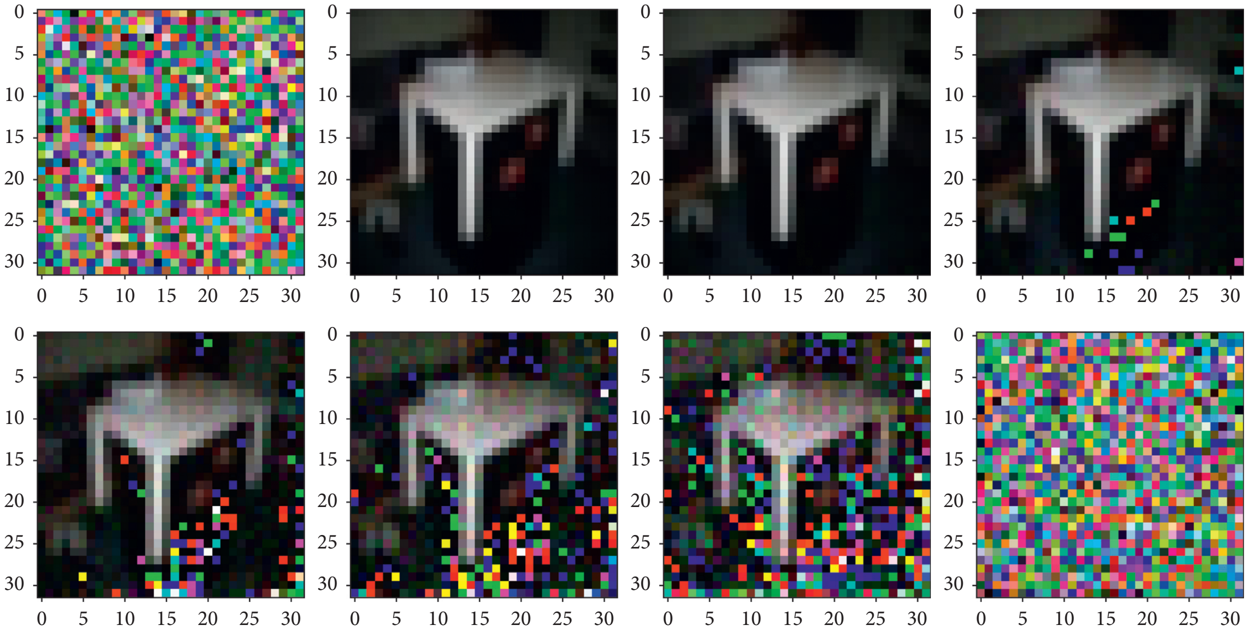


Figure 6: The results of the DLG attacks with the CIFAR100 dataset in different compression.

Table 5: The evaluation metrics of the GAN-based attacks with the MNIST dataset in different compression.

Method	Iterations	CE	Communication frequency	SSIM	MSE
GAN	500	1	1	0.6248	37.3750
GAN	500	0.97	1	0.6138	40.1095
GAN	500	0.95	1	0.6107	45.3234
GAN	500	0.93	1	0.5861	45.6194
GAN	500	0.90	1	0.6144	39.9441

We can draw the following conclusions. (1) Changing communication frequencies is one of the key factors affecting the success in an attack. The more local training iteration, the more difficult it is to be attacked, i.e., the more secure the client data is in the training process of the federated learning system; (2) compressing the weights (parameters) cannot only avoid data leakage but also affect the security of federated learning, and the more the parameters and the smaller the compressing rate, the higher the security. Therefore, the communication frequency and

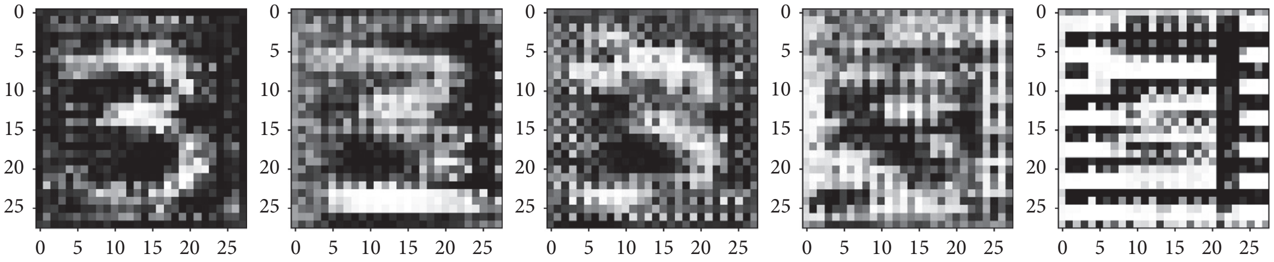


Figure 7: The results of the GAN-based attacks with the MNIST dataset in different compression.

parameter compression are two important factors that affect the security of federated learning. If a single value is changed, it will increase the security, but the quality of the federated learning model will be sacrificed.

4. Adaptive Frequency-Compression Federated Learning

In order to improve the security of the federated learning model and reduce the effect on the quality of the global model, we propose an adaptive frequency-compression federated learning (AFC-FL) by adjusting the communication frequency and parameter compression. The weights of the two factors are adjusted to ensure the accuracy of federated learning adaptively, while providing higher security. This calls for AFC-FL to start from a larger frequency and minimal compression and adjust them gradually as the model reaches closer to convergence. Such an adaptive strategy will offer a win-win in system operation by ensuring communication efficiency and security.

4.1. Adaptive Strategy. This approach of AFC-FL is to change the communication frequency and parameter compression rates in each iteration round, according to the loss value in the model during training. However, the fixed iteration rounds are difficult to be determined without prior knowledge. Therefore, we divide the entire training process into multiple identical iteration rounds. At the beginning of each iteration round, we determine the communication frequency based on the difference between this round and the previous. The parametric compression rate is then affected by the communication frequency. The strategy of AFC-FL is to estimate the choice of two factors accurately and to make the federated learning model more efficient and secure. It will be described in details in the following sections.

During the training phase of federated learning, it is difficult to select the accurate communication period. An alternative is proposed to obtain the basic communication period update rule. Based on this rule, we adjust it to our strategy with fixed iteration rounds. The improved rule is as follows:

$$f_l = \left\lceil \sqrt{\frac{F\left(X_{e=lE_0}\right)}{F\left(X_{e=0}\right)} f_0} \right\rceil, \tag{4}$$

where E_0 is the fixed rounds of updating, $F(X_{e=lE_0})$ is the objective function values of the lth update, and $F(X_{e=0})$ is the initial loss value. The frequency f_l of the next round is guided by the training loss value. When the loss $F(X_{e=lE_0})$ becomes smaller, the frequency f_l decreases, i.e., local computation rounds are fewer, and the communication frequency gradually increases.

It can be concluded that both communication frequency and parameter compression have inhibitory effects on the accuracy and security of federated learning. Although the lower frequency and compression can achieve higher resistance against gradient-based reconstruction attacks, the accuracy will be decreased seriously. There is a need for an adaptive strategy to trade off the accuracy and security. Through the results in Section 2.1, we found that there is a linear relationship between frequency and loss and between compression and loss under a certain constraint: $f \approx k * \text{Loss}$ and $C \approx q * \text{Loss}$, where $0 < f < 5, 90 < C < 100$.

Therefore, we consider whether we can find a balance between frequency, compression, and loss so that the algorithm can guarantee both the accuracy and the security of the model under the joint influence of compression and frequency. We try to find the appropriate Loss to make our algorithm achieve the most effect by giving different values of the Loss interval. We analyze by the following assumptions:

$$\text{Loss} = \frac{1}{k} * f + \frac{1}{q} * C, \tag{5}$$

where Loss is the set constant and k and q are the two influencing factors. The purpose of the formula is to make the obtained communication frequency to influence the parameter compression rate so that the two inhibiting factors do not overlap each other to achieve the effect of adaptive parameter change. Therefore, the formula is organized as follows:

$$C_l = C_0 - \lfloor f_l * D \rfloor, \tag{6}$$

where C_0 is the initial parametric compression rate, C_l is the parametric compression rate after the lth update, and D is the constant used to control the rate of decline. It is found that a low parameter compression rate makes federated learning worse, so we set a minimum threshold for the parameter compression rate ($C_{\min}$). The improved formula is as follows:

$$C_l = \max\left(C_{\min}, C_0 - \lfloor f_l * D \rfloor\right). \tag{7}$$

It has been shown that the communication frequency f_l and the parameter compression C can be adjusted mutually when the appropriate parameter D is set, which affects the completion of the federated learning training and makes the attacks fail.

4.2. Adaptive Communication-Efficient Federated Learning (AFC-FL). To improve the security of the system, we combine multiple influences into a federated learning model, where communication frequency and parameter compression jointly affect the security of the model. Through experimental analysis and research, we propose a method for improving the security of the system, AFC-FL. AFC-FL is comprised of one adaptive frequency model and adaptive compression model. The adaptive frequency model is used to change the frequency by model loss. And, the adaptive compression model is designed to change the parameter compression value by changing the frequency. In the following, we present the network architecture and then analyze the procedure of the distributed optimization.

An overview of the proposed architecture is shown in Figure 8. There are N clients and a central server. The central server aggregates the parameters uploaded by each client. Each client updates parameters according to our proposed AFC-FL.

Algorithm 1 describes the execution process of the AFC-FL algorithm. The initial parameters include the number of clients M, training epochs E_t, updating epochs E_u, optimization function learning rate η, batch size B, and the optimization function is Adam. AdaptFreq is the function of change of the frequency. It can be expressed by equation (4). The function AdaptComp, which can be expressed by equation (7), is to change the compression. Comp can decrease the number of parameters by compression. The algorithm is divided into two parts, client side and server side. The server side is responsible for controlling the global model generation, while the client side performs the adaptive algorithm updates and the local model uploads.

The execution process on the server side is (1) initialize the model $W_{g,0}$; (2) at round i, collect the sparse parameters $W_{m,i}$ uploaded by m clients and find the next round of global model $W_{g,i+1}$ by means of mean aggregation; (3) send the new round of the global model down to each client.

The execution process of the clients is (1) first download the global model $W_{g,i}$ sent by the parameter server; (2) determine whether it is an update interval before each iteration, and if so, perform the update function to update the communication frequency f and the parameter compression ratio C; (3) then, train each client node locally according to the new communication frequency; (4) obtain the locally compressed model $W_{m,i}$ according to the parameter compression ratio C and by compressing the parameters of the locally trained model ω; (5) upload the model $W_{m,i}$ to the server side.

In Algorithm 1, lines 8–12 execute the AFC-FL algorithm after a certain number of rounds through the code, adjusting the communication frequency of the local model as well as the parameter compression rate after the training is completed. When the communication frequency is higher, the more the parameters of the model trained by each client change, the less effective the attacker's attack will be. At this time, 15 lines of parameter compression will not need too much compression to ensure the accuracy of the model training. When the training reaches the late convergence, the communication frequency increases to correct the accuracy and reduce the model upload parameters. Our parameter compression and communication frequency change are calculated on the client side to ensure that the local model parameters are trimmed before uploading. Meanwhile, it avoids joint attacks by the server and the attacker on the client to ensure the security of the system.

5. Experiment Results

To verify the efficiency and security of AFC-FL, we perform experiments using MNIST datasets. In principle, however, AFC-FL can be extended to other types of data, such as medical records. We first show the advantages of our approach by comparing the experiments in Section 3.2; secondly, we perform the GAN-based attacks' experiment to compare the effect of recovered images after the attacks, and we judge the success of our approach by observing the imaging characteristics of the images artificially, combined with the accuracy of the final model.

5.1. Experiment Setup. We mimic the ideas provided by the authors of the GAN article and use Tensorflow to implement the attacks in the privacy scenario of a federated learning client. And, we set up the adaptive frequency parameter compression scheme to further extend in terms of efficiency and security.

5.1.1. Platform. All experiments are completed in the same experimental environment, including Intel (R) Xeon (R) CPU E5-2620 v4 @2.1 GHz, Nvidia 1080Ti GPU (11 GB) ∗2, and 32 GB RAM. Due to the limitation of experimental conditions, the uploading and downloading of shared parameters in the iteration process of the federated model are implemented by the same machine simulation. Obviously, the statistical indicators of the experimental results have nothing to do with the communication method, so the evaluation is still accurate and effective.

5.1.2. Dataset. The MNIST dataset is stored in bytes. The training set contains 60,000 0–9 digital pixel samples and labels, and the test set contains 10,000 0–9 digital pixel samples and labels. Each image is composed of 32×32 pixels. This dataset is one of the deep learning benchmark datasets. For each client in the experiment, we use a non-IID approach, i.e., each client has only one class of images.

5.1.3. Model. We choose to use the architecture of the classical convolutional neural network LeNet5, which is the

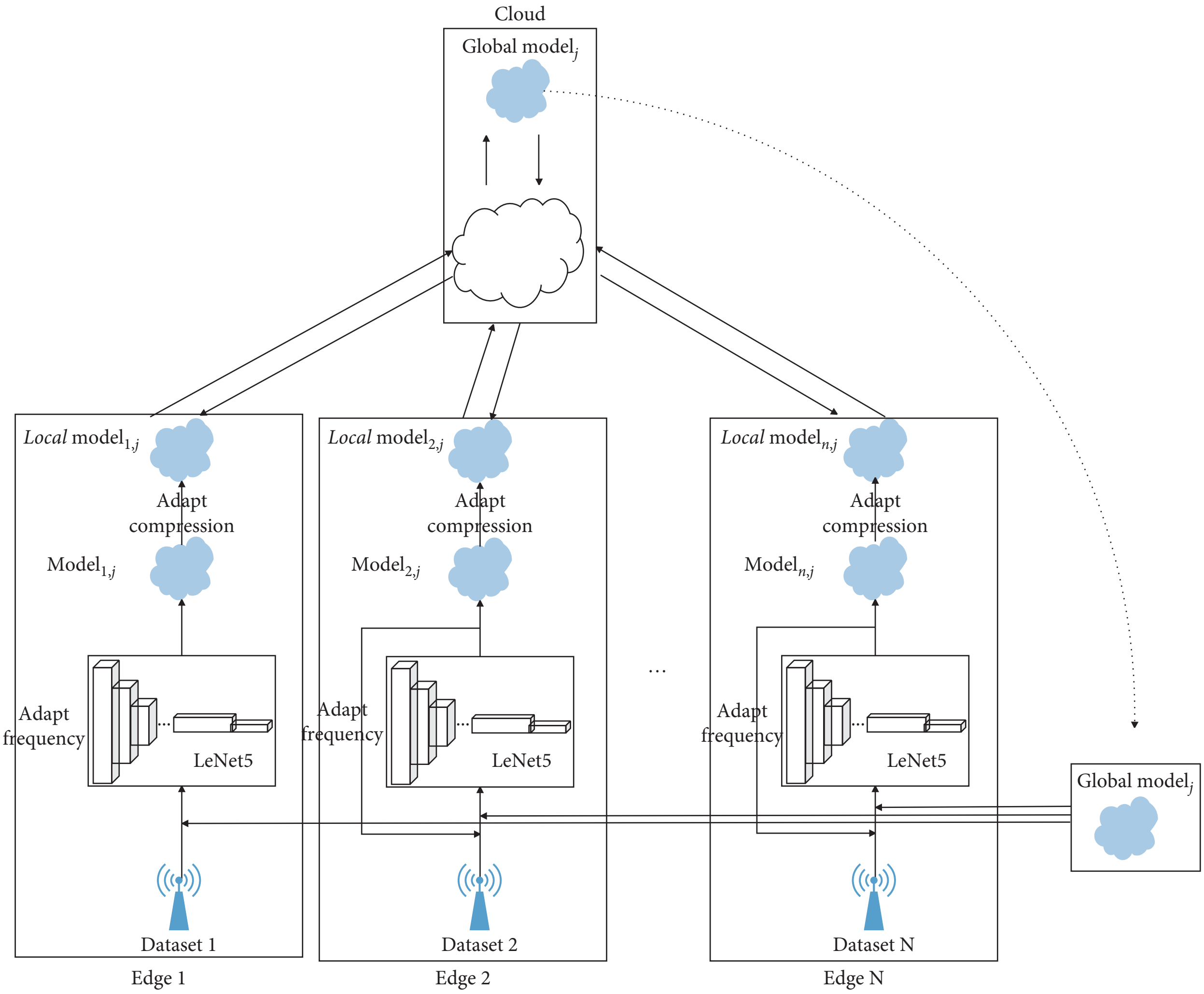


FIGURE 8: A conceptual scheme for the AFC-FL.

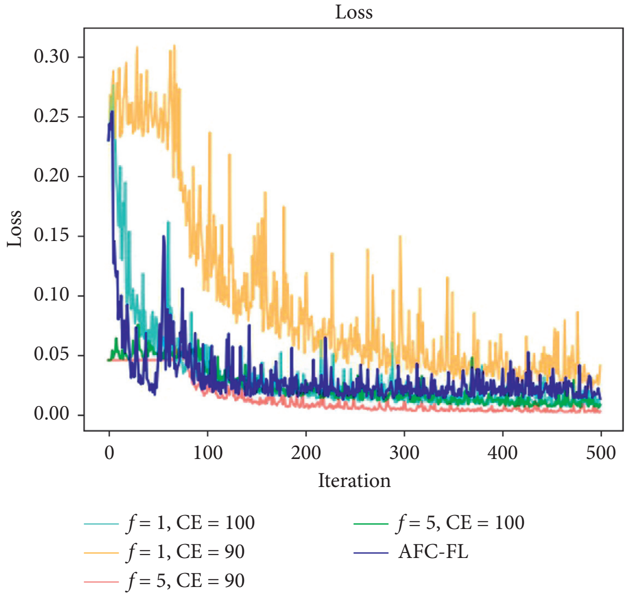


FIGURE 9: The loss of the experimental results.

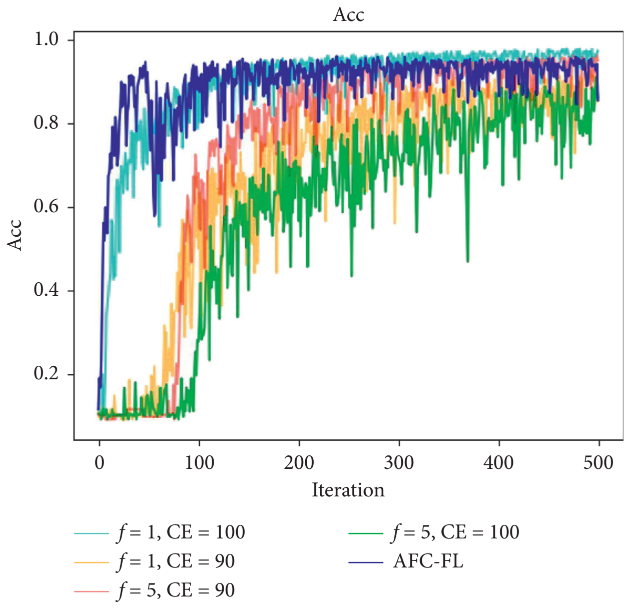


Figure 10: The accuracy of the experimental results.

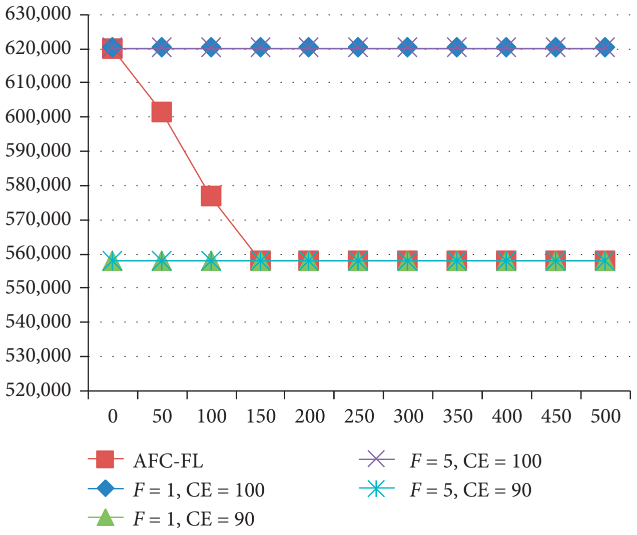


Figure 11: The amount of parameters during training.

basis of many networks such as AlexNet, VGGNet, and ResNet, and it is general with great effect. LeNet5 has seven layers, namely, C1 convolutional layer, S2 pooling layer, C3 convolutional layer, S4 pooling layer, C5 convolutional layer, F6 fully connected layer, and output fully connected layer. Each layer contains trainable parameters; each layer has multiple Feature Maps, and each feature map extracts one feature of the input through a kind of convolutional filter.

5.1.4. Hyperparameter Choice. We choose to use ADAM as optimization algorithms and the batch size of 50; for setting a stable learning rate, we conduct some experiments. In the end, we set the learning rate to 0.01.

5.1.5. Metrics. We compare the performance of proposed AFC-FL with the following methods at a fixed frequency or compression period. (1) Baseline: fully compression and one communication iteration; (2) manually adjust the frequency, i.e., using the same frequency for each iteration and for multiple experimental comparisons; (3) manually tuned the compression case where compression is changed by frequency before new training epochs. We train all methods for a long time to convergence and compare the results of 500 iterations.

5.2. Efficiency Experiment. In our experiments, we evaluate the results using the different learning rates or batch sizes. We find that higher learning rates make the federated

```
Input: The number of clients $M$;
Input: Training epochs $E_t$;
Input: Updating epochs $E_u$;
Input: Learning rate $\eta$;
Input: Local mini-batch size $B$;
Input: Local optimization function Adam;
Output: A global model $W_g$;
Procedure ServerExecute:
    initialize $W_{g,0}$;
    for each Iteration $i \in [1, E_t]$ do
        $W_{g,i+1} \leftarrow 1/M \sum_{m=1}^{M} W_{m,i}$;
    end for
end procedure
Procedure ClientUpdate ($m\ W_{g,i}$):
    for each Iteration $i \in [1, E_t]$ do
        if $i\%E_u == 0$ then
            $f \leftarrow \text{AdaptFreq}(L)$: $f = \left\lceil \sqrt{F(X_{e=iE_u})/F(X_{e=0}) f_0} \right\rceil$
            $C \leftarrow \text{AdaptComp}(f)$: $C = \max(C_{\min}, C_0 - \lfloor f * D \rfloor)$
        end if
        for each Client $m \in [1, M]$ do
            $\omega \leftarrow \text{Adam}(W_{g,i}, f, \eta)$
            $W_{m,i} \leftarrow \text{Comp}(\omega, C)$
        end for
    end for
end procedure
```

Algorithm 1: An adaptive communication-efficient federated learning (AFC-FL).

Table 6: The comparative experiment results.

Function	$\text{avg}_{\text{parameter}}$	acc_{90} (epochs)	The parameters of acc_{90}	$\max_{\text{acc}}$
AFC-FL	570,434	86	5.265×10^7	0.9597
$F = 1$, CE = 100	620,060	98	6.076×10^7	0.9738
$F = 1$, CE = 90	558,042	329	1.836×10^8	0.9166
$F = 5$, CE = 100	620,060	221	1.370×10^8	0.9622
$F = 5$, CE = 90	558,039	459	2.561×10^8	0.9092

learning model unstable and suffer from model oscillations, and smaller batch sizes result in worse convergence of the model. After analysis of several results, we set a learning rate of 0.01 and a batch size of 50 as the hyperparameter. Meanwhile, we also conduct several experiments on the variation range of the adaptive frequency and parameter compression rate. It shows that the frequency is greater than 5, and the parameter information uploaded by each client node is more vague, which makes the federated learning less effective; when the parameter compression rate is lower than 90, the loss of critical information of the local model will also lead to the reduction of the quality of the federated learning model. Therefore, we set thresholds to control the frequency and parameter compression range when using AFC-FL.

We compare the results between the AFC-FL function and the threshold set at a critical value after the communications of 500 epochs. From Figure 9, we can find that our method converges nearly as fast as the comparative experiments' scheme. In Figure 10, the accuracy of our method is better than the experiment that the frequency is 5 and no matter whether it has parameter compression or not. Although the experimental accuracy which with a frequency of 1 and no parameter compression is slightly higher than ours, the cost of communication is much higher than ours, and our method provides more security.

We use the cost of communication as an evaluation criterion, i.e., the number of uploader parameters in the same communication round determines the upload time of the local model to the server, which affects the efficiency of the global model. The amount of client traffic handled by the server in the same round can be used to represent the throughput of the federated training system. The user participation will be low in a bandwidth-constrained communication environment. Our algorithms make the system to allow more users to participate in training at the same time by reducing the number of uploaded parameters, which improves the throughput. In order to succinctly compare the cost of communication, we use the average number of parameters uploaded in each communication round as the evaluation criterion. The number of parameters uploaded at each epoch is

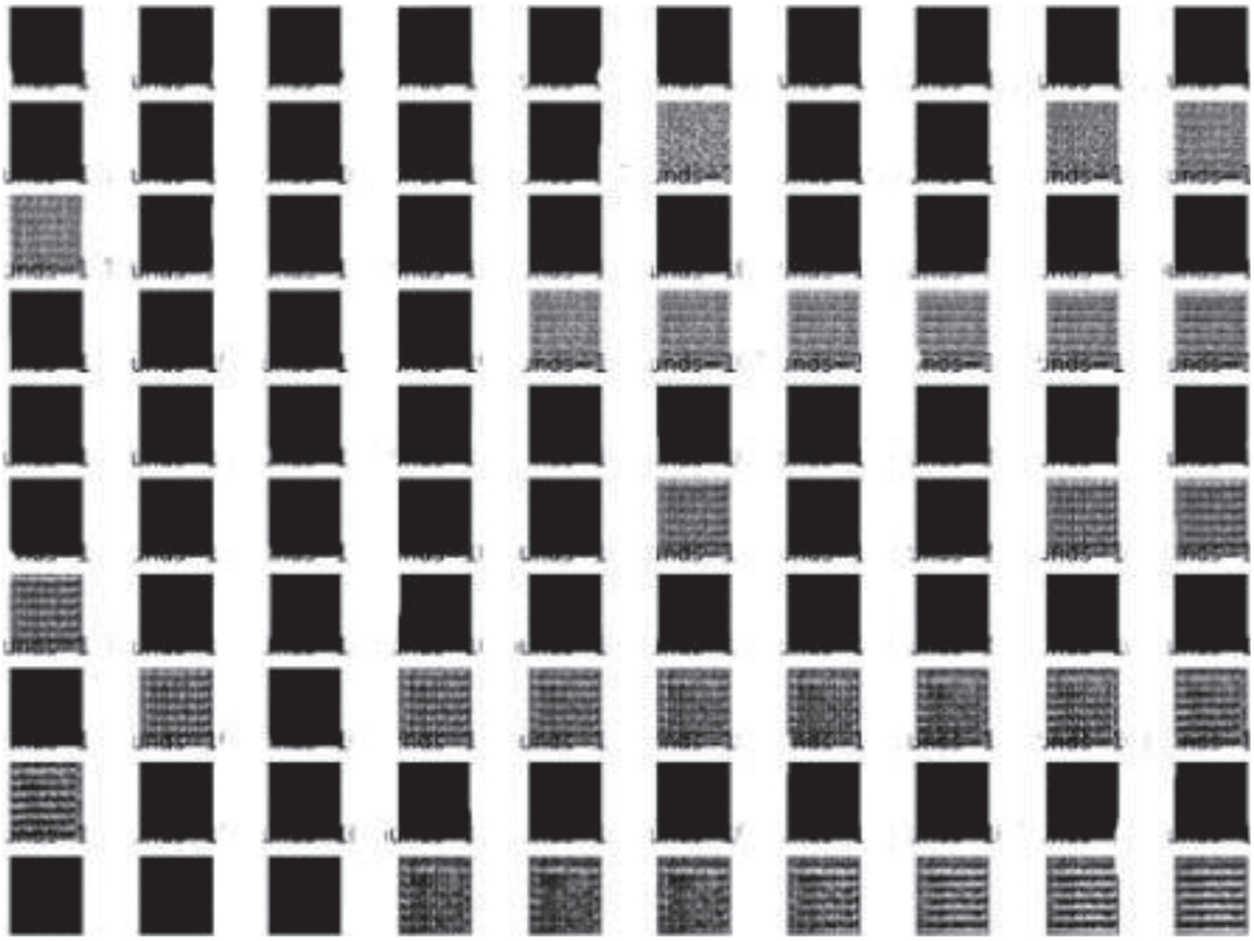


Figure 12: The reconstruction results of GAN-based attack on AFC-FL.

shown in Figure 11. We record the epoch number when the accuracy of model achieving 90%. In addition, we also focus the accuracy after 500 rounds.

From Table 6, we can find that AFC-FL uses the fewest epochs to achieve 90% accuracy for the first time, where $avg_{parameter}$ is the mean parameters of 500 epochs' communication, acc_{90} is the accuracy reached 90% for the first time, the parameters of acc_{90} are the total parameters of the accuracy reached 90% for the first time, and max_{acc} is the maximum of the accuracy in 500 epochs. We verify that the compression and frequency can impact the global model to achieve high accuracy in Section 3. Thus, the results of comparative experiments prove the superiority of our method. Since the model also has the compression to ensure the security of the federated learning model, the accuracy is slightly reduced, but the small reduction in accuracy is acceptable in exchange for the improvement in communication efficiency and the security of the whole system.

5.3. Security Experiment. We now evaluate the security of our AFC-FL against GAN-based attacks. We partition the MNIST dataset into 10 clients by numbers 0–9, with each client having only one of the numerical datasets. We preprocess each client before the formal iteration to avoid any failure to converge due to obscure model features. We use LeNet5 as the generator (G) and the model training network, and we perform the training using Algorithm 1. We observe the generative effect in Figure 12, showing the reconstruction results of every five rounds of attacks during 500 iteration rounds. We can find that the picture cannot be

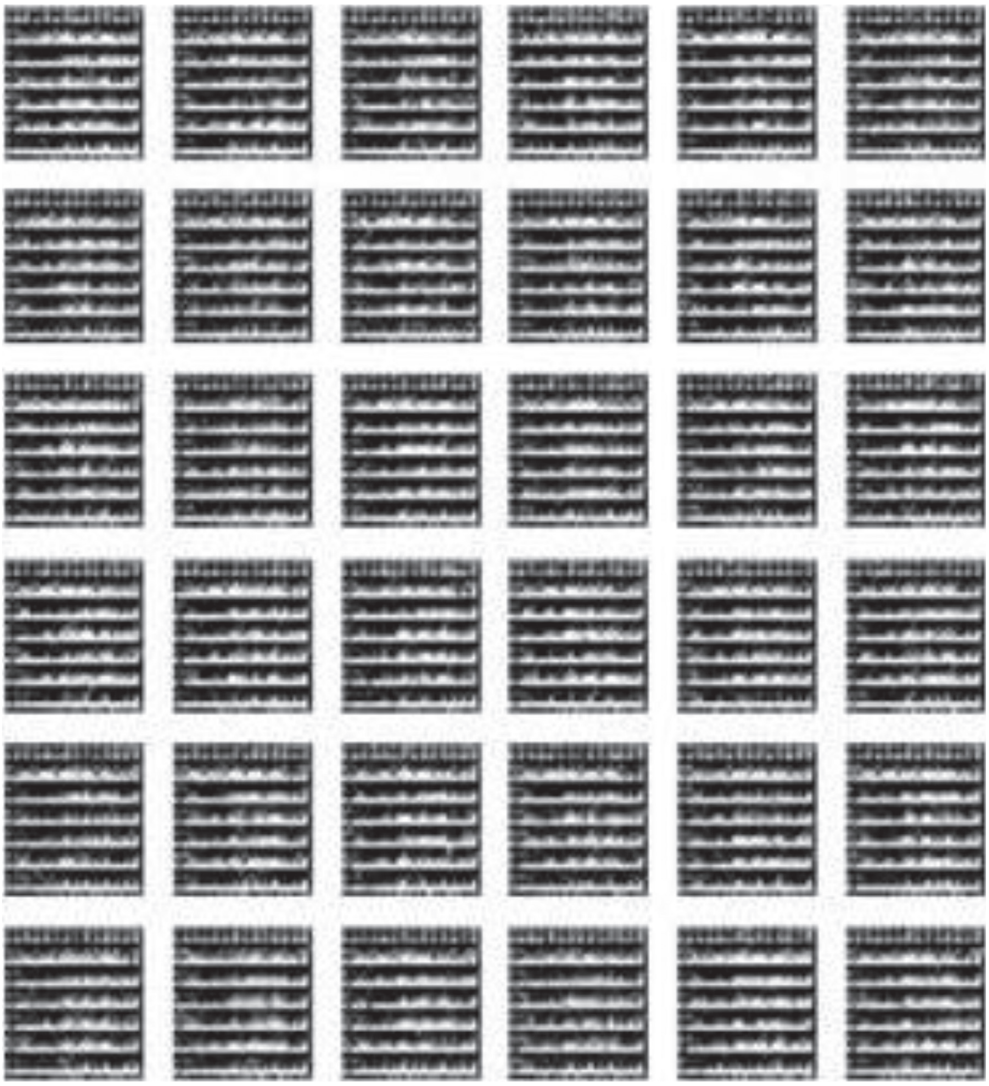


Figure 13: The reconstructed images generated by the GAN-based attacks on AFC-FL.

reconstructed in most cases. It is also unclear which numbers are actually identified. Figure 13 shows the reconstructed image generated by our method after performing 500 rounds of attacks. The generated image is not recognizable to the human eye, and it can be assumed that our method is effective.

6. Conclusions

In this paper, we propose a federated learning optimization algorithm (AFC-FL) with adaptive frequency and compression selection in IoT. The sparsification or communication delay technique significantly reduces the communication cost for clients, improving the security during gradient transmission. Meanwhile, the adaptive strategy also decreases the communication costs of clients. Verified analysis of the algorithms and experimental results using MNIST datasets conclude that AFC-FL is effective in resisting gradient-based reconstruction attacks. Extensive experiments are conducted to verify the effects of resisting attacks and communication time of our algorithm compared to fixed frequency or fixed compression. Experimental results show that AFC-FL not only significantly reduces the communication traffic but also keeps the client data safe to increase the security of the federated learning model, while preserving the convergence. Future works can also consider asynchronous collection of the client parameter, as well as the selection of different update strategies for each client depending on the size of the parameter. It is the goal of our future research studies to ensure security, while speeding up the convergence rate of the model. In addition, we may also consider improvements in homomorphic encryption and differential privacy. How to improve the efficiency using a low additional overhead is also important to research.

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 work was supported by the National Natural Science Foundation of China under Grant no. 62072247.

References

[1] F. Mármol, C. Sorge, O. Ugus, and G. Pérez, "Do not snoop my habits: preserving privacy in the smart grid," *IEEE Communications Magazine*, vol. 50, no. 5, pp. 166–172, 2012.

[2] Z. Tian, C. Luo, J. Qiu et al., "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 16, pp. 1963–1971, 2019.

[3] Z. Tian, W. Shi, Y. Wang et al., "Real-time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4285–4294, 2019.

[4] Z. Guan, Y. Zhang, L. Zhu, L. Wu, and S. Yu, "Effect: an efficient flexible privacy-preserving data aggregation scheme with authentication in smart grid," *Science China Information Sciences*, vol. 62, Article ID 32103, 2019.

[5] J. Geiping, H. Bauermeister, H. Drge et al., "Inverting gradients–how easy is it to break privacy in federated learning?," 2020, http://arxiv.org/abs/2003.14053.

[6] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in Neural Information Processing Systems*, pp. 14774–14784, 2019.

[7] B. Zhao, K. R. Mopuri, and H. Bilen, "IDLG: Improved deep leakage from gradients," 2020, http://arxiv.org/abs/2001.02610.

[8] B. Hitaj and G. A. F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 603–618, Dallas, TX, USA, October 2017.

[9] Z. Wang, M. Song, Z. Zhang et al., "Beyond inferring class representatives: user-level privacy leakage from federated learning," in *Proceedings of the 2019-IEEE Conference on Computer Communications IEEE INFOCOM*, pp. 2512–2520, Paris, France, 2019.

[10] W. Cheng, W. Ou, X. Yin et al., "A privacy-protection model for patients," *Security and Communication Networks*, vol. 2020, Article ID 6647562, 12 pages, 2020.

[11] O. Goldreich, "Secure multi-party computation," *Manuscript Preliminary Version*, vol. 78, 1998.

[12] B. Ghazi, R. Pagh, and A. Velingker, "Scalable and differentially private distributed aggregation in the shuffled model," 2019, http://arxiv.org/abs/1906.08320.

[13] K. Bonawitz, V. Ivanov, B. Kreuter et al., "Practical secure aggregation for federated learning on user-held data," 2016, http://arxiv.org/abs/1611.04482.

[14] M. A.A. Chu et al., "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, New York, NY, USA, October 2016.

[15] Q. Li, Z. Wu, Z. Wen, and B. He, "Privacy-preserving gradient boosting decision trees," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 784–791, 2020.

[16] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *Proceedings of the 2013 IEEE Global Conference on Signal and Information Processing*, pp. 245–248, Austin, TX, USA, 2013.

[17] M. Asad, A. Moustafa, and T. Ito, "FedOpt: towards communication efficiency and privacy preservation in federated learning," *Applied Sciences*, vol. 10, no. 8, pp. 1–17, 2020.

[18] P. Han, S. Wang, and K. K. Leung, "Adaptive gradient sparsification for efficient federated learning: an online learning approach," 2020, http://arxiv.org/abs/2001.04756.

[19] Y. Zhou, X. Chen, and M. Chen, "Privacy-preserving multidimensional data aggregation scheme for smart grid," *Security and Communication Networks*, vol. 2020, Article ID 8845959, 14 pages, 2020.

[20] J. Wang and G. Joshi, "Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 212–229, 2019.

[21] W. Wei, L. Liu, M. Loper et al., "A framework for evaluating gradient leakage attacks in federated learning," 2020, http://arxiv.org/abs/2004.10397.

[22] N. Strom, "Scalable distributed DNN training using commodity GPU cloud computing," in *Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association*, Dresden, Germany, September 2015.

[23] A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," 2017, http://arxiv.org/abs/1704.05021.

[24] L. Tian, Z. Liu, V. Sekar et al., "Privacy for free: communication-efficient learning with differential privacy using sketches," 2019, http://arxiv.org/abs/1911.00972.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6631075, 21 pages
https://doi.org/10.1155/2021/6631075

Research Article

An Efficient Communication Intrusion Detection Scheme in AMI Combining Feature Dimensionality Reduction and Improved LSTM

Guanyu Lu and Xiuxia Tian

College of Computer Technology and Science, Shanghai University of Electric Power, Shanghai 200090, China

Correspondence should be addressed to Xiuxia Tian; xxtian@shiep.edu.cn

Received 17 December 2020; Revised 13 February 2021; Accepted 22 March 2021; Published 21 April 2021

Academic Editor: Jinguang Han

Copyright © 2021 Guanyu Lu and Xiuxia Tian. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Communication intrusion detection in Advanced Metering Infrastructure (AMI) is an eminent security technology to ensure the stable operation of the Smart Grid. However, methods based on traditional machine learning are not appropriate for learning high-dimensional features and dealing with the data imbalance of communication traffic in AMI. To solve the above problems, we propose an intrusion detection scheme by combining feature dimensionality reduction and improved Long Short-Term Memory (LSTM). The Stacked Autoencoder (SAE) has shown excellent performance in feature dimensionality reduction. We compress high-dimensional feature input into low-dimensional feature output through SAE, narrowing the complexity of the model. Methods based on LSTM have a superior ability to detect abnormal traffic but cannot extract bidirectional structural features. We designed a Bi-directional Long Short-Term Memory (BiLSTM) model that added an Attention Mechanism. It can determine the criticality of the dimensionality and improve the accuracy of the classification model. Finally, we conduct experiments on the UNSW-NB15 dataset and the NSL-KDD dataset. The proposed scheme has obvious advantages in performance metrics such as accuracy and False Alarm Rate (FAR). The experimental results demonstrate that it can effectively identify the intrusion attack of communication in AMI.

1. Introduction

In recent years, as the Internet of Things (IoT) technology is commonly used in the power industry, Smart Grid has become the development direction of future power grids. The core architecture of Smart Grid connecting with the computer network is AMI. AMI is a complicated system directly related to electricity consumption information, privacy information, and electricity transaction information. The possible threat of network intrusion has a huge impact on the reliable operation of the Smart Grid [1, 2]. As an influential research content of network communication security, intrusion detection has been widely discussed by experts and scholars. The application of intrusion detection algorithms represents one of the research hotspots in the field of communication in AMI in recent years. Radoglou-Grammatikis and Sarigiannidis [3] summarized the contribution of intrusion detection and prevention system (IDPS) to the Smart Grid paradigm and provided an analysis of 37 cases. Intrusion detection can be viewed as a classification problem, using machine learning algorithms and data mining algorithms to classify network data into normal traffic and intrusion attack traffic [4]. When the intrusion detector finds misbehavior, it can take appropriate actions immediately so that any harm to the system will be minimized [5]. At present, related research can be divided into misuse-based detection [6] and anomaly-based detection [7] according to detection technology. The misuse-based intrusion detection scheme matches the extracted network traffic with the data traffic, which has the existing type tags. If the detected traffic and intrusion attack traffic have similar characteristics, the system will send out an alarm message. Such a method has good performance in identifying existing attacks by establishing a pattern library of intrusion attacks.

However, as an emerging model in the Smart Grid, there will be many new types of attacks appearing in AMI. The accuracy of the Misuse-based intrusion detection has decreased significantly, so it cannot meet the existing needs of the communication environment in AMI.

By judging the degree of deviation among the features of the collected traffic and the normal traffic, the anomaly-based intrusion detection scheme identifies intrusion attacks. It can be divided into intrusion detection based on statistical learning [8], traditional machine learning [9], and deep learning [10]. Because of the requirements of data distribution, intrusion detection methods based on statistical learning have been eliminated. With the development of Artificial Intelligence (AI) technology, the accuracy of methods based on machine learning has been significantly improved. However, communication traffic in AMI presents the characteristics of large data volume, high-dimensional data, and complex feature information. Methods based on traditional machine learning have the limitation of manually setting features in feature selection. Such methods can only be applied to simple and shallow learning. By constructing a deep hierarchical network structure, the methods based on deep learning can learn advanced features from data automatically. This method saves time for feature engineering [11]. The experimental result shows that Autoencoder (AE) has satisfactory performance in feature dimensionality reduction, and LSTM has an exceptional ability to solve classification problems. Currently, researchers have proposed a variety of intrusion detection schemes based on these two models. Dong et al. [12] proposed an intrusion detection model named AE-AlexNet based on deep learning. They use AE to realize dimensionality reduction of high-dimensional traffic. This method fails to achieve layer-by-layer training for high-dimensional traffic, and the robustness of the model is relatively poor. Due to the availability of LSTM on time series data, Althubiti et al. [13] proposed an intrusion detection scheme based on LSTM. In this scheme, only the unidirectional structural features are extracted, and it cannot determine important features. It has serious limitations in its application.

To deal with the above problems, we propose a communication intrusion detection scheme by combining feature dimensionality reduction and improved LSTM in AMI. The main contributions of this paper can be summarized as follows:

(1) First, we propose a Stacked Autoencoder method to achieve feature dimensionality reduction for the high-dimensional features of data in AMI. By extracting the key point information of attributes, it can reduce the calculation time and improve the efficiency of communication intrusion detection in AMI. SAE can modularize and improve the robustness of the neural network.

(2) Second, for the problem that LSTM cannot extract bidirectional structural features, we propose a Bidirectional Long Short-Term Memory model for the classification of traffic. It can reduce the high FAR due to data imbalance in AMI.

(3) Third, to determine the criticality of the dimensionality and the feature, we improve the classification model by the Attention Mechanism. It sets weight coefficients to allocate more attention to key dimensions and important features, to realize accurate detection.

(4) Fourth, the proposed method is compared with the methods based on traditional machine learning and the recent papers on intrusion detection. The experimental results indicate that our scheme is preferable to other competing schemes.

The rest of this paper is organized as follows: Section 2 summarizes the related research work. Section 3 presents the basic theory required in the scheme. Section 4 introduces our intrusion detection scheme in detail. Section 5 is the analysis and comparison of the experiment. Section 6 reports possible threats to the validity of the scheme. The paper is concluded in Section 7.

2. Related Work

This section discusses two types of related work: methods based on traditional machine learning and methods based on deep learning. According to the requirements of communication intrusion detection in AMI, both methods face certain challenges. The challenge of methods based on traditional machine learning is whether the selected features are appropriate for the classification model. Methods based on deep learning face the challenges of high-dimensional features of communication traffic and data imbalance in AMI.

2.1. Intrusion Detection Based on Traditional Machine Learning. Most of the previous research studies are based on traditional machine learning methods, such as Naive Bayes, Decision Tree, *K*-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Hidden Markov Model (HMM). Farid et al. [14] proposed a learning algorithm of intrusion detection, which uses the methods of Naive Bayes and Decision Tree to reconstruct the data. The purpose was to reduce the noise and eliminate the redundant attributes in the training data. This scheme improved the accuracy of different types of network intrusion attacks. Radoglou-Grammatikis and Sarigiannidis [15] proposed an intrusion detection system in AMI based on the CART decision tree, and the system was tested on the intrusion detection dataset CICIDS2017. Accuracy can reach 99.66%, and True Positive Rate (TPR) can reach 99.30%. Senthilnayaki et al. [16] used the Rough Sets Attribute Reduction algorithm to select features and data, extracted the more important features and data, and realized the feature dimensionality reduction of the data attributes in the dataset. Finally, the improved KNN classifier was utilized to complete the classification of traffic, which effectively reduces the FAR of detecting intrusion attacks. For the threats in AMI, Vijayanand et al. [17] constructed an intrusion detection system in Smart Grid through the SVM classifier method. The feature was selected by mutual information value. Finally, through simulation experiments, the detection accuracy of normal records obtained could reach 93.4%, and the detection accuracy of intrusion attack records could reach 89.2%. The advantage of the SVM model is its brilliant generalization ability. The FAR of the final results is low, but the SVM model is only suitable for solving the binary classification problem. For detecting multiclass intrusion attacks in AMI, the performance of the SVM model is not good. Hurley et al. [18] used HMM algorithm to develop an adaptive network intrusion detection system, which had a superior performance in detecting intrusion attacks in Software Defined Network (SDN). High-quality training datasets can be constructed through the clustering algorithm in traditional machine learning. This kind of method can make the type of the dataset from complex to simple. It can reduce the spatial dimension of the data and the computational overhead. In previous studies, clustering algorithms used in intrusion detection include *k*-means algorithm [19], hierarchical clustering [20], and Principal Component Analysis (PCA) [21]. Unsupervised machine learning technology is an imperative method for data processing and feature engineering in the context of massive data in AMI. However, such algorithms are sensitive to the outliers and noise of data.

The arrival of the big data era indicates that intrusion detection has entered a stage of large data volume, high data dimension, high network bandwidth, and complex feature information. Methods based on traditional machine learning need to manually set features, which are relatively shallow learning methods. It is therefore difficult to achieve the purpose of prediction and analysis.

2.2. Intrusion Detection Based on Deep Learning. The deep learning method constructs a network structure constituted by multiple hidden layers to adapt to the higher-dimensional learning process by learning the internal laws and representation levels of sample data. At the stage of feature engineering, the convergence time of the model is saved. The deep learning algorithms commonly used in the field of communication intrusion detection in AMI include Autoencoder, Recurrent Neural Network (RNN) and its excellent variants, and Convolutional Neural Network (CNN). To obtain hidden information, Sun et al. [22] adopted the idea of the Variational Autoencoder (VAE) to achieve feature dimensionality reduction. They can extract more advanced features than manually set features. The proposed scheme has shown good performance on the KDD-CUP dataset, Mnist dataset, and UCSD pedestrian's dataset. Distributed Denial of Services (DDoS) is one of the most notorious attacks in AMI. Learning features through a multilayer AutoEncoder, Ali and Li [23] proposed an efficient DDoS attack detection technique. Bhardwaj et al. [24] combined stacked sparse AutoEncoder and Deep Neural Network (DNN) to detect DDoS attacks in cloud computing. However, there are various types of intrusion attacks in AMI, which cannot guarantee the robustness of the scheme. Gao et al. [25] proposed a new intrusion detection method based on the LSTM model, which can be used in Supervisory Control And Data Acquisition (SCADA) systems. Agarap [26] introduced a linear SVM to replace the *softmax* function in the final output layer of the Gated Recurrent Unit (GRU) and built a model named GRU-SVM for intrusion detection. Finally, through simulation experiments, they showed the superiority of their scheme in training and testing time. Roy and Cheung [27] proposed to detect attacks based on the BiLSTM model in IoT. They used the UNSW-NB15 dataset for testing and achieved an accuracy of over 95% in attack detection. Khan et al. [28] built the intrusion detection system consisting of two stages: The first stage is the anomaly detection module based on Spark-ML, and the second stage is the misuse detection module based on Convolutional-LSTM (Conv-LSTM). In the cross-validation, the accuracy rate can reach 97.29%. Riyaz and Ganapathy [29] used CNN to select the most contributory feature and classify the traffic when identifying and detecting intrusion attacks on wireless networks. The proposed intrusion detection system achieved an overall accuracy of 98.88%. Lin et al. [30] proposed a framework named IDSGAN by Generative Adversarial Networks (GAN) to generate adversarial attacks to deceive and evade intrusion detection systems. Based on the NSL-KDD dataset, experiments had proved the feasibility of this model to attack systems that can detect multiple different attacks, and had achieved excellent results.

Although intrusion detection based on deep learning has many advantages in feature learning, it also has some shortcomings. During the communication of AMI, the training dataset contains a large number of normal data samples, and the proportion of intrusion attack traffic is very small. The records of data show the phenomenon of unbalance. Meanwhile, the structural data are high-dimensional, and the features need to be selected and extracted. In response to the need for communication intrusion detection in AMI, this paper proposes a corresponding scheme to solve the above problems.

3. Preliminary: Basic Theory

This section introduces the basic theory of the model used in the next section.

3.1. Autoencoder. Autoencoder is an unsupervised neural network algorithm, which can reconstruct the vector which is input into the model [31]. It shows powerful nonlinear generalization capabilities and has been applied in many fields, such as image denoising [32] and anomaly detection [33]. Figure 1 is the hierarchical structure of the Autoencoder. The Autoencoder consists of two parts: encoder (visible layer to hidden layer) and decoder (hidden layer to output layer). The encoder is represented by the function $h = f_{\theta}(x)$, and it maps the input data to the feature space. The decoder is represented by the function $x' = g_{\theta'}(h)$, and it maps the encoded data back to the sample space for the generation and reconstruction of the input vector.

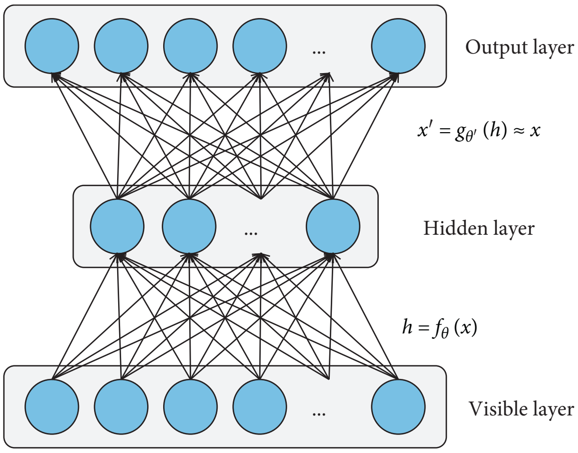


Figure 1: The hierarchical structure of the Autoencoder. The structure consists of a visible layer, a hidden layer, and an output layer. The visible layer and the hidden layer constitute the encoder part. The hidden layer and the output layer constitute the decoder part.

The learning target of an Autoencoder is to make the input equal to the output. The objective function of the network is $g_{\theta'}(f(x)) \approx x$, which means learning an identity. We suppose that the set of data samples input to the Autoencoder model is $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. The training dataset formed after removing the category labels is $\{x_1, x_2, \ldots, x_n\}$.

The encoder can be expressed by the following equation:

$$h_k = f_\theta(x_k) = R\left(\sum_{i=1}^{n_1} W_{ij} \cdot x_{ki} + b_i\right). \tag{1}$$

The decoder can be expressed by the following equation:

$$x'_k = g_{\theta'}(h_k) = R\left(\sum_{i=1}^{n_2} W_{ij} \cdot h_{ki} + b_i\right). \tag{2}$$

The *loss* function can be expressed by the following equation:

$$L_{W,b}(x_k, x'_k) = \frac{1}{2}\sum_{i=1}^{n_1}(x_k - x'_k)^2. \tag{3}$$

In equations (1)–(3), x_k is the kth training sample. h_k is the value of each neuron in the hidden layer of the kth training sample. x'_k is the value of each neuron in the output layer of the kth training sample. n_1 is the number of neurons in the visible layer, and n_2 is the number of neurons in the hidden layer. W_{ij} is the connection weight of the ith neuron in the previous layer and the *jth* neuron in the next layer. b_i is the bias term of the ith neuron on the corresponding layer. R indicates that the activation function used in the Autoencoder is the *Relu* function.

For the task of feature dimensionality reduction, the final output x'_k and the original data x_k have the same feature dimensions, which is meaningless. The result of h_k in the hidden layer is the dimensionality reduction expression of x_k. It is obtained without losing the original data information as much as possible. We can accomplish the target of transforming high-dimensional features into low-dimensional features.

3.2. Long Short-Term Memory. Long Short-Term Memory neural network was proposed as an improved variant of Recurrent Neural Network, which mainly solves the problems of gradient disappearance or explosion that may occur during RNN training. It is more suitable for use in sequence data processing with long-term correlation [34]. Figure 2 displays the LSTM network structure.

Figure 3 is the internal structure of the memory storage unit of LSTM, which is mainly composed of the forget gate, the input gate, and the output gate [35]. The forget gate is responsible for processing the output of the previous layer, selecting useful information, and filtering useless information. The input gate is responsible for judging the importance of information and updating the status of the unit through critical information. The output gate is responsible for determining which unit status can be input to the unit of the next layer.

The forget gate can be expressed by the following equation:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right). \tag{4}$$

In the equation, the value of h_{t-1} and x_t is 0 or 1. After the forget gate, if the output is 0, the current useful information will be stored. And if the output is 1, the current useless information will be deleted.

The calculation process of the input gate consists of two parts: one is to determine the important information that needs to be added to the unit status through the *sigmoid* activation function, and the other is to use the tanh activation function to form a new vector to update the unit status. The equations of the two parts are shown in the following equations:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{5}$$

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C). \tag{6}$$

At this time, the original unit status C_{t-1} is updated to C_t, and the equation is as follows:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t. \tag{7}$$

The output gate determines the output through the *sigmoid* function, which can be expressed by the following equations:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \tag{8}$$

$$h_t = o_t * \tanh(C_t). \tag{9}$$

In equations (4)–(9): σ indicates that the activation function used is *sigmoid*. h_{t-1} is the hidden layer status of the previous layer unit. W_f is the weight of the forget gate, and b_f is the bias term of the forget gate. W_i is the weight of the input gate, and b_i is the bias term of the input gate. W_C is the weight of the unit status, and b_C is the bias term of the unit

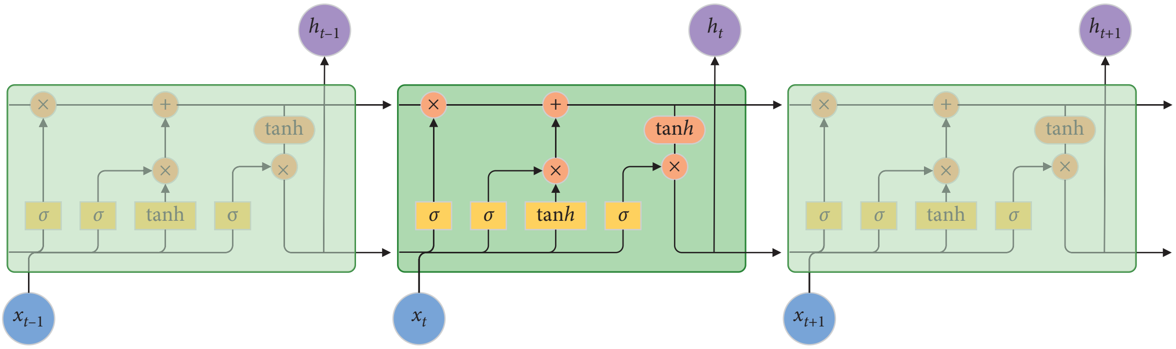


FIGURE 2: The framework of the LSTM model. *Sigmoid* and tanh activation functions are used inside the unit.

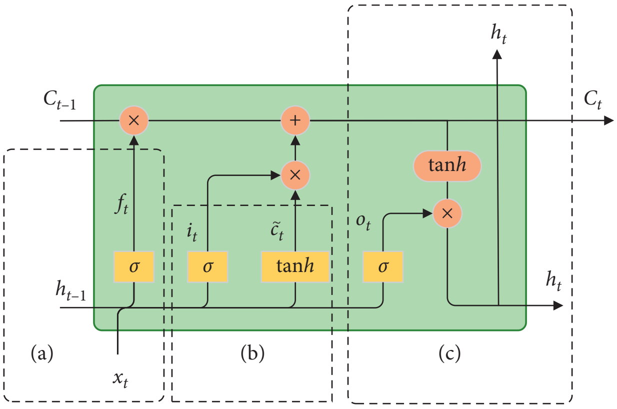


FIGURE 3: The internal structure of the LSTM unit. (a) Forget gate. (b) Input gate. (c) Output gate.

status. W_o is the weight of the output gate, and b_o is the bias term of the output gate.

4. Proposed Method

The intrusion detection scheme we proposed is mainly for the communication scenario in AMI, and Figure 4 provides a detailed description of this scenario. AMI is generally composed of smart meters, concentrators, grid servers of measurement management, and its communication network. The bottom component of AMI is the smart meter. It is responsible for collecting and analyzing user information on Home Area Network (HAN), Business Area Network (BAN), and Industry Area Network (IAN), while monitoring and recording electricity consumption data and other statistical data. The intermediate component of the system is the data collector deployed in the Neighbor Area Network (NAN), responsible for summarizing the data information received from the smart meter. The top component of the system is the device of AMI headend, which is deployed in the Wide Area Network (WAN) and is responsible for collecting data from multiple data collectors. There are multiple feasible communication methods and protocols at each level. For example, the ZigBee protocol stack and Bluetooth communication are used in the HAN. In the NAN, the communication standard of WiFi is used. There are numerous communication methods in the WAN, such as Digital Subscriber Line (DSL), optical fiber communications, and GPRS communication. Figure 4 also captures the collection environment of communication data in AMI. The Programmable Logic Controller (PLC) and the data acquisition unit are connected to the communication server on the data bus through the switch. On the data bus, potential intrusion attack threat terminals send abnormal traffic during the communication, carry out various attacks, and affect the communication between normal devices. The database server is responsible for storing communication log files between devices. The data acquisition unit is in charge of collecting normal traffic and intrusion attack traffic.

In this section, we first introduce the data preprocessing approach. Then, we describe the proposed communication intrusion detection model in AMI and explain how to classify the records of normal traffic and intrusion attack traffic.

4.1. Data Preprocessing. The communication traffic in the AMI system contains many nondigital features. Such features cannot be directly used as input to the model, so it is necessary to convert nondigital features into digital features. We apply the idea of one-hot encoding to process data and use n-bit status registers to encode n states of nondigital features. Assume that nondigital features have n states such as {Status_1, Status_2, . . ., Status_n}, and Table 1 is the final result of the encoding.

To eliminate the difference of the feature quantification results and prevent the features with a large value range from affecting the model results, we use the procedure of normalization for all features to process the obtained data. Normalization of data can improve the accuracy of the model and speed up the solution of the model. In our proposed scheme, the Min-Max normalization method is used [36]. Assuming that the dataset of a group of features is $\{X_1, X_2, \ldots, X_n\}$, the equation for normalizing a certain data X_i in the set is shown in the following equation:

$$X_i' = \frac{X_i - X_{\text{Min}}}{X_{\text{Max}} - X_{\text{Min}}}. \quad (10)$$

Here, X_i' is the normalized data and X_{Min} and X_{Max} are the minimum and maximum values in the dataset, respectively.

The pseudocode description of the algorithm at this stage is shown in Algorithm 1.

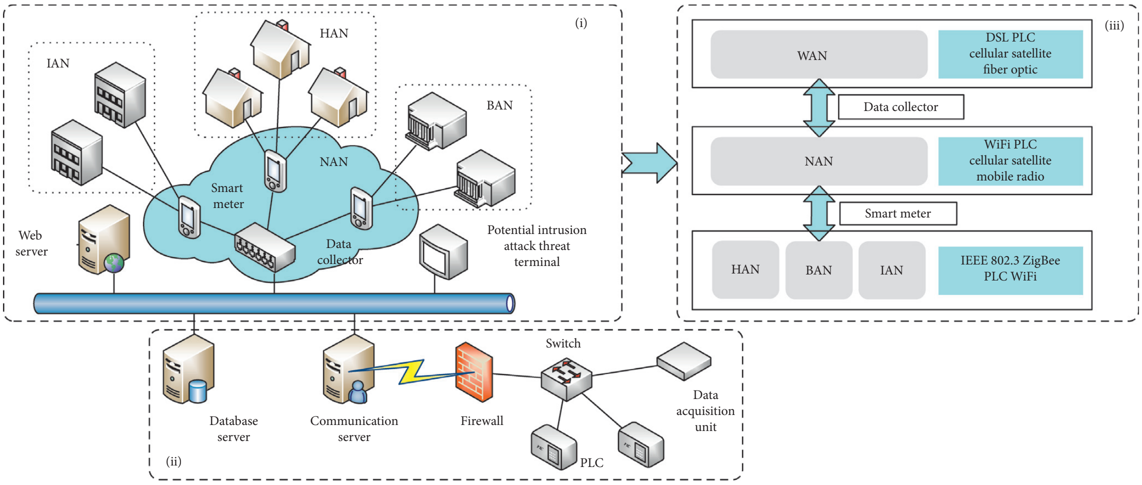


Figure 4: Communication scenario in AMI. (i) Communication network hierarchy structure in AMI. (ii) The collection environment of communication data in AMI. (iii) Protocols used in each layer of AMI.

Table 1: Encoding results of n states of nondigital features.

Description of the status	One-hot encoding result (n bits)
Status_1	(0, 0, . . ., 0, 1)
Status_2	(0, 0, . . ., 1, 0)
. . .	. . .
Status_n−1	(0, 1, . . ., 0, 0)
Status_n	(1, 0, . . ., 0, 0)

4.2. Intrusion Detection Model. The preprocessed data can be used for classification detection. Figure 5 is the established communication intrusion detection model in AMI, which is divided into two parts: the first part is to perform dimensionality reduction operations on the features of the data through the Stacked Autoencoder, which is marked by (i) in Figure 5. The second part is to classify the traffic through the improved LSTM for the data after feature reduction, to achieve the purpose of identifying intrusion attacks. This part is indicated by (ii) in Figure 5.

4.2.1. SAE for Feature Dimensionality Reduction. The Stacked Autoencoder is a neural network made up of multiple layers of sparse Autoencoders, which can effectively extract features and reduce feature dimensions [37]. Figure 6 is the structure of the Stacked Autoencoder. This SAE is composed of n Autoencoders stacked. Hidden layers (1~n−1) are the output of the previous Autoencoder and the input of the next Autoencoder. In this case, one Autoencoder is nested inside another, and learning takes place in a layer-by-layer greedy learning manner [38]. After every Autoencoder is trained, the decoder part is removed, and the final target output is attached to the innermost encoder layer.

To use the AMI dataset for communication traffic, after completing the data preprocessing, the vector with a dimension of 196 is obtained. This vector is used as the input of SAE. Figure 7 is the design of SAE to achieve feature selection and dimensionality reduction. In the proposed scheme, the number of hidden layers of the SAE network structure is 4 layers. The number of neurons in the hidden layer is {128, 64, 32, 32}, and finally, a 32-dimensional vector is selected as the output.

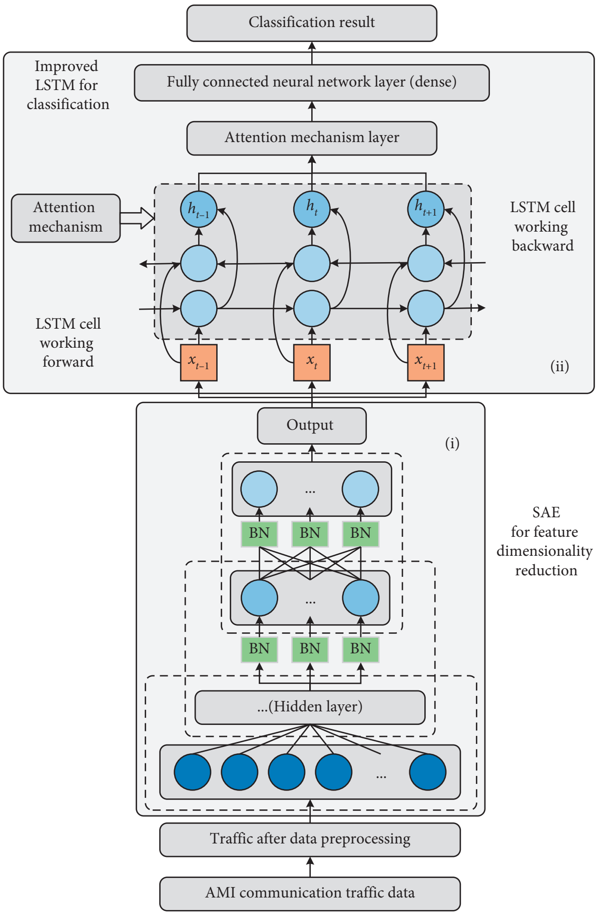

Classification result
Improved LSTM for classification
Fully connected neural network layer (dense)
Attention mechanism layer
Attention mechanism
h_{t-1}
h_t
h_{t+1}
LSTM cell working backward
LSTM cell working forward
x_{t-1}
x_t
x_{t+1}
(ii)
Output
(i)
BN
...(Hidden layer)
SAE for feature dimensionality reduction
Traffic after data preprocessing
AMI communication traffic data

Figure 5: Communication intrusion detection model in AMI. The overall model has two stages: (i) SAE for feature dimensionality reduction. (ii) Improved LSTM for classification.

The processes of SAE for feature dimensionality reduction generally include two stages: pretraining and fine-

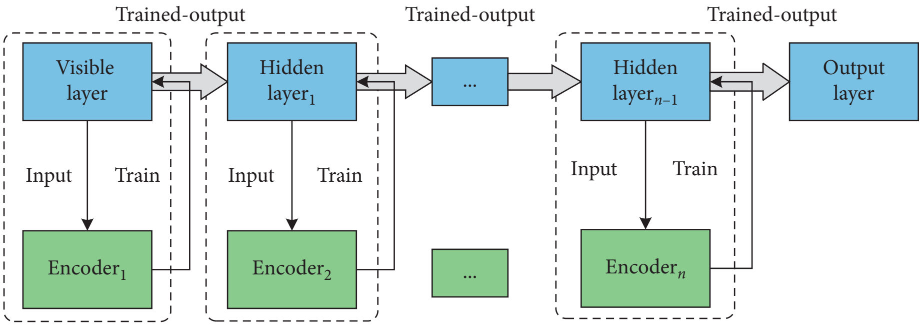


FIGURE 6: The structure of SAE. The output of the Autoencoder of the previous layer is used as the input of the Autoencoder of the next layer.

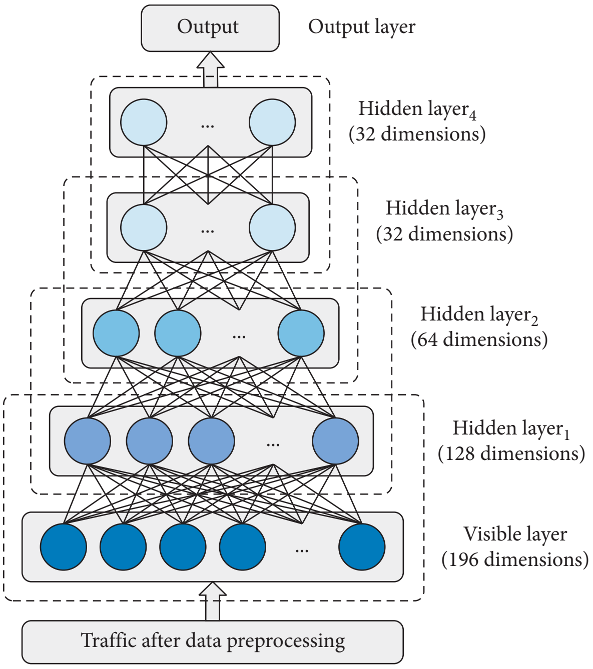

Output
Output layer
Hidden layer$_4$ (32 dimensions)
Hidden layer$_3$ (32 dimensions)
Hidden layer$_2$ (64 dimensions)
Hidden layer$_1$ (128 dimensions)
Visible layer (196 dimensions)
Traffic after data preprocessing

FIGURE 7: The process of feature selection and dimensionality reduction implemented by SAE in the proposed scheme. The structure of the hidden layer in SAE is {128,64,32,32}.

tuning. Pretraining is an unsupervised training process that uses a large amount of unlabeled traffic data to perform layer-by-layer greedy learning and training in SAE. The steps of the pretraining stage are as follows:

Step 1: input the preprocessed AMI communication data into the visible layer of the SAE, and initialize the connection weight W and bias term b randomly.

Step 2: train the network parameters of the Hidden layer$_1$ and calculate the output of the Hidden layer$_1$ through the trained parameters.

Step 3: use the unsupervised learning method to train the Autoencoder and calculate the *loss* function value $L_{W,b}$. Keep updating the weight W and the bias term b until the *loss* function value finally reaches the set threshold and no longer changes.

Step 4: use the output of the previous network layer as the input of the next network layer and apply the same method to train the parameters of this hidden network layer. Repeat Step 3 until all layers of the SAE have been trained.

The above pretraining process cannot obtain a mapping from the input communication traffic in AMI to the output label, so one or more connection layers need to be added to the last layer of the SAE network, and the backpropagation method is used for training. This stage is called the fine-tuning process. The steps of the fine-tuning stage are as follows:

Step 1: construct the entire SAE by connecting the hidden layers trained by each AE, and set the connection weight W and the bias term b to the values obtained in the pretraining stage.

Step 2: cascade a *softmax* classifier after the last layer, and train the network parameters of the *softmax* classifier in combination with the labeled original data.

Step 3: use the network parameters of the pretraining stage and the fine-tuning stage as the initialization parameters of the entire deep network. Find the parameter values around the minimum value of the cost function as the optimal parameter.

Step 4: use the backpropagation algorithm to fine-tune the optimal parameters obtained in the SAE model.

The pseudocode of the algorithm using the SAE to complete feature dimensionality reduction is shown in Algorithm 2.

The scheme has been put forward since the process of completing feature dimensionality reduction. As the network deepens, the training process becomes more difficult and the convergence speed becomes slower. We adopted the idea of Batch Normalization (BN) [39] to solve this problem. In Step 4 of the fine-tuning stage, there is the phenomenon that gradient disappears in the low-layer neural network during backpropagation. BN means that the input value distribution of any neuron in each layer of the neural network is forced back to the standard normal distribution with the mean of 0 and the variance of 1 through the normalization method. In this way, the input value of the nonlinear transformation function falls into an area that is sensitive to the input, to avoid the problem of vanishing gradient. Figure 8 displays the process of improving the hidden layer network

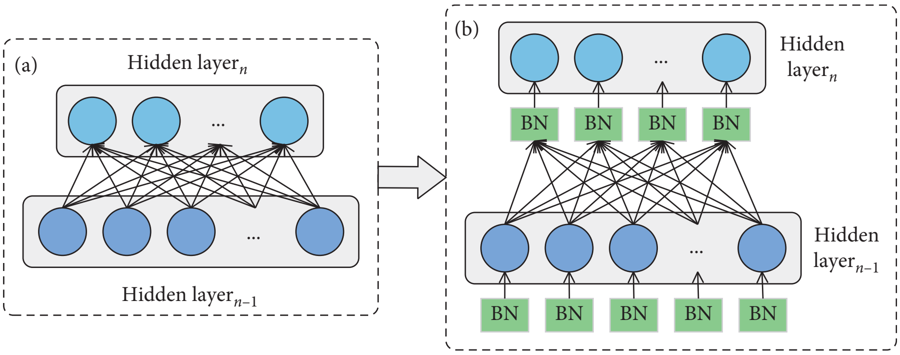


FIGURE 8: Using the BN method to improve the hidden layer network structure of the Autoencoder. (a) The original network structure between two layers. (b) The improved network structure between two layers.

structure of the Autoencoder. We use the idea of BN for processing behind each hidden layer.

In the BN operating experience, the equation for transforming the activation value of each neuron in the hidden layer is shown in the following equation:

$$\widehat{x}^{(k)} = \frac{x^{(k)} - E\left[x^{(k)}\right]}{\sqrt{\operatorname{Var}\left[x^{(k)}\right]}}. \tag{11}$$

In the equation, $x^{(k)}$ is the linear activation value of the corresponding neuron in this layer. $E[x^{(k)}]$ is the average value of linear activation values obtained by all training instances in this training process. $\operatorname{Var}[x^{(k)}]$ is the variance of the linear activation value. Assuming that there are n instances in the training process, the calculation equations of $E[x^{(k)}]$ and $\operatorname{Var}[x^{(k)}]$ are

$$E\left[x^{(k)}\right] = \frac{1}{n}\sum_{i=1}^{n} x_i^{(k)}, \tag{12}$$

$$\operatorname{Var}\left[x^{(k)}\right] = \frac{1}{n}\sum_{i=1}^{n}\left(x_i^{(k)} - E\left[x^{(k)}\right]\right)^2. \tag{13}$$

To prevent the expression ability of the SAE network from decreasing after changing the distribution, the adjustment parameters *scale* and *shift* are added at each neuron to activate the inverse transformation operation. The equation for the inverse operation is shown in the following equation:

$$y^{(k)} = \gamma^{(k)}\widehat{x}^{(k)} + \beta(k). \tag{14}$$

4.2.2. Improved LSTM for Classification. After completing data dimensionality reduction, it is necessary to classify the normal traffic and intrusion attack traffic of communication data in AMI. The scheme in this paper applies the Bi-directional Long Short-Term Memory [40] method. Figure 9 is the structure of the BiLSTM model.

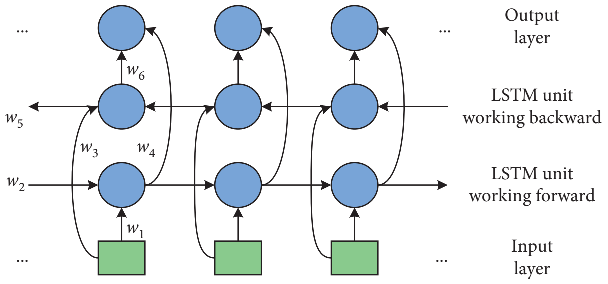


FIGURE 9: The structure of the BiLSTM network. It consists of a forward working LSTM and a backward working LSTM.

In the proposed scheme, the input layer is responsible for sequence encoding of the data after feature dimensionality reduction. The forward working LSTM unit is responsible for extracting the forward features of the data sequence in the input layer, and the backward working LSTM unit is responsible for extracting the backward features of the data sequence in the input layer. The output layer integrates the data output by the forward and backward transmission layers.

The calculation equations inside the LSTM unit of forward transmission are shown in the following equations:

$$\vec{f}_t = \sigma\left(\vec{W}_f \cdot \left[\vec{h}_{t-1}, \vec{x}_t\right] + \vec{b}_f\right), \tag{15}$$

$$\vec{i}_t = \sigma\left(\vec{W}_i \cdot \left[\vec{h}_{t-1}, \vec{x}_t\right] + \vec{b}_i\right), \tag{16}$$

$$\vec{C}_t = \vec{f}_t * \vec{C}_{t-1} + \vec{i}_t * \tanh\left(\vec{W}_C \cdot \left[\vec{h}_{t-1}, \vec{x}_t\right] + \vec{b}_C\right), \tag{17}$$

$$\vec{o}_t = \sigma\left(\vec{W}_o\left[\vec{h}_{t-1}, \vec{x}_t\right] + \vec{b}_o\right), \tag{18}$$

$$\vec{h}_t = \vec{o}_t * \tanh\left(\vec{C}_t\right). \tag{19}$$

The calculation equations inside the LSTM unit of backward transmission are shown in the following equations:

$$\overleftarrow{f}_t = \sigma\left(\overleftarrow{W}_f \cdot \left[\overleftarrow{h}_{t-1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_f\right), \tag{20}$$

$$\overleftarrow{i}_t = \sigma\left(\overleftarrow{W}_i \cdot \left[\overleftarrow{h}_{t-1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_i\right), \tag{21}$$

$$\overleftarrow{C}_t = \overleftarrow{f}_t * \overleftarrow{C}_{t-1} + \overleftarrow{i}_t * \tanh\left(\overleftarrow{W}_C \cdot \left[\overleftarrow{h}_{t-1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_C\right), \tag{22}$$

$$\overleftarrow{o}_t = \sigma\left(\overleftarrow{W}_o \left[\overleftarrow{h}_{t-1}, \overleftarrow{x}_t\right] + \overleftarrow{b}_o\right), \tag{23}$$

$$\overleftarrow{h}_t = \overleftarrow{o}_t * \tanh\left(\overleftarrow{C}_t\right). \tag{24}$$

The output vector h_t of the output layer can be calculated from the output vectors $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ of the forward and backward hidden layers, respectively. The calculation equation is shown in the following equation:

$$h_t = \left[\overrightarrow{h}_t \oplus \overleftarrow{h}_t\right]. \tag{25}$$

In equation (25), $\oplus$ is the combination method of forward and backward output vectors.

In the communication intrusion detection scheme in AMI, to pay corresponding attention to the different features of the intrusion attack traffic data, the Attention Mechanism is introduced. This will improve the accuracy of intrusion detection. Attention Mechanism is widely used in image processing [41], natural language processing [42], target detection [43], and other fields. The core idea of this method is to imitate the way the human body observes objects and select more critical parts from a large amount of information to achieve the purpose of feature extraction. The Attention Mechanism is used in two measures in the improved LSTM to classify traffic data. One is to use the Attention Mechanism to determine which dimensions play a critical role in classification. The other is that the data sequence results obtained by the BiLSTM output layer are added to the Attention Mechanism layer to obtain a more accurate classification. The calculation equations are shown in the following equations:

$$u_t = \tanh\left(W_w h_t + b_w\right), \tag{26}$$

$$a_t = \operatorname{soft}\max\left(u_t^T, u_w\right), \tag{27}$$

$$v = \sum a_t h_t. \tag{28}$$

In equations (26)–(28), u_t is the attribute representation of the output vector h_t of the BiLSTM output layer, a_t is the weight of importance, v is the result of the importance weighting operation on the output vector h_t, and u_w is a randomly generated context vector during training.

The final result is input to the fully connected neural network layer for classification, and the prediction result is obtained. The pseudocode of the algorithm for implementing traffic data classification using improved LSTM is shown in Algorithm 3.

5. Experimental Results and Analysis

This section first introduces the experimental environment and the dataset used. Then, we compare with other methods and debug the internal structure and parameters of the model to illustrate the superiority of the proposed scheme of communication intrusion detection scenarios in AMI.

5.1. Experimental Settings and Dataset Description. The experiment was run on a machine with Windows 10 operating system, Intel Core i9-9900K CPU, NVIDIA RTX2080 Ti GPU, and 32 GB RAM. To compare the running time of the proposed model on CPU and GPU, we conducted the comparative experiment on a machine with Windows 10 operating system with Intel Core i7-5500U CPU and 8 GB RAM. The methods mentioned in the scheme are all programmed with Python 3.7, and the compiler used is Pycharm2020. A large number of programming libraries in python are employed in programming, such as Numpy, Pandas, Keras, and Sklearn. Numpy provides the basic packages for data analysis and high-performance scientific computing, which can calculate the matrix precisely and work with vectors. Pandas is a tool based on Numpy, including different libraries and various standard data types. It is used to accurately process large-scale datasets. Keras is the most important library in the process of programming, and it can run on TensorFlow or Theano. Our scheme is running on TensorFlow. Because some deep learning models (AE, LSTM) are used in the designed scheme, Keras provides a flexible deep learning framework for it. We can easily and quickly implement the scheme programming through Keras. Sklearn encapsulates a large number of machine learning algorithms, such as classification, regression, and clustering.

To evaluate our proposed communication intrusion detection scheme in AMI, the public intrusion detection standard dataset UNSW-NB15 is employed for verification. This dataset was created by the cyber security research group of the Australian Centre for Cyber Security (ACCS) [44], which addresses the issue of data redundancy in other datasets. The traffic obtained by the AMI system has the characteristic of more normal data and less intrusion attack data. An unbalanced dataset is needed to verify the proposed scheme. Table 2 contains the distribution of the data. The UNSW-NB15 dataset has 175341 records in the training dataset and 82332 records in the testing dataset. In addition to normal data records, the dataset has 9 types of intrusion attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. There are 93000 normal records in the dataset, accounting for 36.09%. However, Shellcode and Worms have 1511 records and 174 records, respectively, accounting for 0.59% and 0.07%. Therefore, the UNSW-NB15 dataset meets the verification characteristics of the unbalanced dataset.

```
 Input: original training dataset Original_train, testing dataset Original_test
Output: preprocessed training dataset Preprocessed_train, testing dataset Preprocessed_test
train = pd.read_csv (Original_train)
test = pd.read_csv (Original_train)
/ ∗  concat() complete data splicing  ∗ /
Spliced_data = pd.concat([train, test])
/ ∗  get_dummies() complete one-hot encoding  ∗ /
Encoded_data = get_dummies(Spl_data, [“Feature_1”, “Feature_2”, . . ., “Feature_n”])
Encoded_data.drop([“label”, “attack_cat”])
/ ∗ MinMaxScaler() normalizes the data to [0, 1]  ∗ /
Preprocessed_train = MinMaxScaler (Encoded_data, train, feature_range = (0, 1))
Preprocessed_test = MinMaxScaler (Encoded_data, test, feature_range = (0, 1))
End
```

Algorithm 1: Algorithm description of data preprocessing stage.

```
 Input: Preprocessed_train, Preprocessed_test, Train_label, Test_label
Output: Encoded_train, Encoded_test, Train_label, Test_label
Load processed data Preprocessed_train, Preprocessed_test
While not reach terminating condition: n-layer autoencoder training (n = 1, 2, 3, 4)
  for Epoch in range (1, 100):
    / ∗  complete filepath stitching  ∗ /
    os.path.join()
    / ∗  Save the model results after each epoch to filepath  ∗ /
    AE_n_point = ModelCheckpoint (filepath, monitor = “val_loss”, verbose = 1, save_best_only = True, mode = “min”)
    / ∗  Save the best model to prevent overfitting  ∗ /
    AE_n_stops = EarlyStopping (monitor = “val_loss”, patience = 10, mode = “min”)
    break
  AutoEncoder_n.load_weights()
  Output the prediction result of this layer: layer_n_output, test_n_out
End While
Encoded_train = SAE_encoder.predict (train)
Encoded_test = SAE_encoder.predict (test)
/ ∗  save SAE final result  ∗ /
np.save (Encoded_train, Encoded_test, Train_label, Test_label)
End
```

Algorithm 2: Using the SAE to implement feature dimensionality reduction algorithm.

Network communication technology is applied to AMI, and massive data present a multidimensional characteristic structure. The UNSW-NB15 dataset comprises 44 features [45], which like the characteristic of communication traffic with many features in AMI. Table 3 displays all the features and types of the dataset. These features are divided into 6 categories:

Flow features: it includes the data flow attributes that the communication terminal interacts with. In this dataset, only proto belongs to Flow Features, which is used to mark the transaction protocol used in the communication.

Base features: it shows the basic attributes of the traffic in the records, such as the features of the protocol connection.

Content features: it is related to the attributes of the TCP and the HTTP.

Time features: it includes all the time attributes of the data in the record, such as the arrival time of the data packet, the return confirmation time, and the survival time.

Additional generated features: it can be divided into two parts: one is general feature attributes, and the other is connection feature attributes. In general feature attributes, each feature gets its use from the defense point of view. Connection feature attributes only provide defenses in connection attempts.

Labeled features: it indicates whether the record is normal data or generated from an intrusion attack. Both the normal and attack records are marked with a Boolean type.

5.2. Performance Evaluation Metrics. We are required to set model evaluation standards to test the effectiveness of the designed communication intrusion detection scheme in

```
Input: Encoded_train, Encoded_test, Train_label, Test_label
Output: Classification result
Load feature reduced data Encoded_train, Encoded_test, Train_label, Test_label
Define parameters time_steps, batch_size
Train_label_ = np. insert (Train_label)
Test_label_ = np.insert (Test_label)
train_generator = TimeseriesGenerator (Encoded_train, Train_label_)
test_generator = TimeseriesGenerator (Encoded_test, Test_label_)/ ∗ define Attention Mechanism function ∗ /
def attention_3d_block (inputs)/ ∗ define Attention Mechanism function ∗ /
lstm1 = Bidirectional (LSTM (units = 24)) (input_traffic) / ∗ define the first layer lstm ∗ /
Call the attention_3d_block () function to judge the criticality of the dimension
lstm2 = Bidirectional (LSTM (units = 12)) (attention_output) / ∗ define the second layer lstm ∗ /
Input BiLSTM output layer results into Attention Mechanism layer
mlp = Dense (units = 6, activation = "relu") (attention _output2)
mlp2 = Dense (units = 1, activation = "sigmoid") (mlp) / ∗ The fully connected neural network layer outputs the classification results ∗ /
for Epoch in range(1, 250):
    history = classifier.fit_generator (train_generator, steps_per_epoch, callbacks = [], validation_data = test_generator, validation_steps)
np.save (Classification result) /∗ save classification final result ∗/
End
```

Algorithm 3: Use improved LSTM for data classification.

AMI. Intrusion attack traffic detection is a classification problem, and its performance metrics depend on the confusion matrix [46]. Table 4 enumerates the definition and explanation of each item in the confusion matrix.

The confusion matrix can be used to calculate the following performance metrics to evaluate our proposed scheme:

Accuracy can be used to represent the proportion of all traffic data (normal and intrusion attack) being classified correctly. The calculation equation is shown in the following equation:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}. \tag{29}$$

Precision can be used to express the probability that the data detected as a positive sample is truly a positive sample. The calculation equation is shown in the following equation:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{30}$$

Recall can express the ratio of intrusion attack traffic detected as positive samples by our proposed scheme to the overall intrusion attack traffic. *Recall* can be used as an important performance metric in the detection of datasets with an unbalanced category. The calculation equation is shown in the following equation:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{31}$$

F1_score is the harmonic value of *Precision* and *Recall*, and the calculation equation is shown in the following equation:

$$F1\_\text{score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{32}$$

FAR reports the ratio of the normal traffic detected as a positive sample to the overall normal traffic. The calculation equation is shown in the following equation:

$$\text{FAR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{33}$$

5.3. Experimental Results. According to our proposed communication intrusion detection scheme in AMI, the UNSW-NB15 dataset is used for verification. First, we preprocess the dataset and use one-hot encoding to convert the nondigital features in the dataset into digital features. Then, we input the preprocessed data into the Stacked Autoencoder for feature dimensionality reduction. The encoded data are input into the improved LSTM model for classification. Finally, the result of classification is output through the fully connected layer. Table 5 is the confusion matrix obtained from the detection results of the proposed scheme on the UNSW-NB15 dataset. Table 6 comprises the results of performance metrics calculated by the values of items in the confusion matrix.

5.4. Comparison with Other Methods

5.4.1. Comparison with Traditional Machine Learning Methods. To prove the advantages of our proposed AMI communication intrusion detection scheme, traditional machine learning methods are utilized to classify and detect the UNSW-NB15 dataset. Table 7 compares their final results with the results of our proposed scheme. Traditional

Table 2: The distribution of records in the UNSW-NB15 dataset.

Number	Record class	Size	Distribution (%)
1	Normal	93000	36.09
2	Fuzzers	24246	9.41
3	Analysis	2677	1.04
4	Backdoors	2329	0.90
5	DoS	16353	6.35
6	Exploits	44525	17.28
7	Generic	58871	22.85
8	Reconnaissance	13987	5.43
9	Shellcode	1511	0.59
10	Worms	174	0.07
	Totals	257673	100

machine learning methods are not particularly effective for intrusion detection on the dataset with unbalanced categories. The best methods about the results are Random Forest and Decision Tree. The performance metric of Accuracy can reach 0.8583 and 0.8531, respectively. However, the FAR of these two traditional machine learning methods is very high. Support Vector Machines show good performance on the Binary Classification problems. After being used for the detection of intrusion attack traffic, the FAR value of the SVM model can reach 0.0079, but the classification Accuracy is only 0.6486. In summary, compared with traditional machine learning schemes, our proposed communication intrusion detection scheme in AMI greatly improves the Accuracy of intrusion detection, while ensuring low FAR. Especially, for datasets with unbalanced categories, it has better performance of classification.

5.4.2. Comparison with Recent Intrusion Detection Scheme. Table 8 compares our proposed scheme with some recent intrusion detection schemes. DO_IDS [47] proposed an intrusion detection algorithm that relies on mixed data optimization. The Time-related NIDS [48] scheme uses a time-related deep learning method to detect intrusion attacks in the network. The SDAE + SVM [49] scheme also uses Denoising Autoencoder (DAE) to reduce the feature dimension. But, different from the scheme we proposed, this scheme finally uses the idea of SVM for classification.

Table 8 suggests that our proposed communication intrusion detection scheme in AMI can better detect intrusion attack traffic compared to the recently proposed papers of intrusion detection. Reference [48] used the time-series model for the scheme. Reference [49] used the feature dimensionality reduction of the Denoising Autoencoder. The scheme we propose is to use an improved LSTM to classify time series data based on the feature dimensionality reduction of SAE. So, we compare the accuracy and loss of each Epoch on the training dataset and the testing dataset of the proposed model and the two deep learning methods. Figure 10 is the curve drawn founded on the results obtained. The three models were trained with 180 Epochs, and the values of accuracy and loss after each Epoch were recorded. The proposed scheme converges significantly faster during training and testing, and the final classification accuracy of our model is considerably higher.

Table 3: Features in the UNSW-NB15 dataset.

Number	Feature	Feature type	Data type
1	proto	Flow features	Nominal
2	state	Base features	Nominal
3	dur	Base features	Float
4	sbytes	Base features	Integer
5	dbytes	Base features	Integer
6	sttl	Base features	Integer
7	dttl	Base features	Integer
8	sloss	Base features	Integer
9	dloss	Base features	Integer
10	service	Base features	Nominal
11	sload	Base features	Float
12	dload	Base features	Float
13	spkts	Base features	Integer
14	dpkts	Base features	Integer
15	swin	Content features	Integer
16	dwin	Content features	Integer
17	stcpb	Content features	Integer
18	dtcpb	Content features	Integer
19	smeansz	Content features	Integer
20	dmeansz	Content features	Integer
21	trans_depth	Content features	Integer
22	res_bdy_len	Content features	Integer
23	sjit	Time features	Float
24	djit	Time features	Float
25	rate	Time features	Float
26	sintpkt	Time features	Float
27	dintpkt	Time features	Float
28	tcprtt	Time features	Float
29	synack	Time features	Float
30	ackdat	Time features	Float
31	is_sm_ips_ports	Additional generated features	Binary
32	ct_state_ttl	Additional generated features	Integer
33	ct_flw_http_mthd	Additional generated features	Integer
34	is_ftp_login	Additional generated features	Binary
35	ct_ftp_cmd	Additional generated features	Integer
36	ct_srv_src	Additional generated features	Integer
37	ct_srv_dst	Additional generated features	Integer
38	ct_dst_ltm	Additional generated features	Integer
39	ct_src_ltm	Additional generated features	Integer
40	ct_src_dport_ltm	Additional generated features	Integer
41	ct_dst_sport_ltm	Additional generated features	Integer
42	ct_dst_src_ltm	Additional generated features	Integer
43	attack_cat	Labeled features	Nominal
44	Label	Labeled features	Binary

To compare the computational cost of the proposed scheme and other schemes, Figure 11 reports the running time for each parameter in the training dataset and the testing dataset to reach the set threshold. We use GPU to

Table 4: Explanation of the meaning of each item in the confusion matrix.

Items in confusion matrix	Explanation
TP	The number of intrusion attack traffic detected as positive samples
TN	The number of normal traffic detected as negative samples
FP	The number of normal traffic detected as positive samples
FN	The number of intrusion attack traffic detected as negative samples

Table 5: Confusion matrix over the UNSW-NB15 dataset using SAE + Attention – BiLSTM.

		Predicted	
		Anomalous	Normal
Actual	Anomalous	36275	313
	Normal	172	45160

accelerate the training speed of all models. Although the proposed scheme spends more time on training for each Epoch, less Epoch is used to reach the threshold. Considering the overall running time, we can complete intrusion detection faster in AMI. The time for the power system to build up a defense mechanism has been extended.

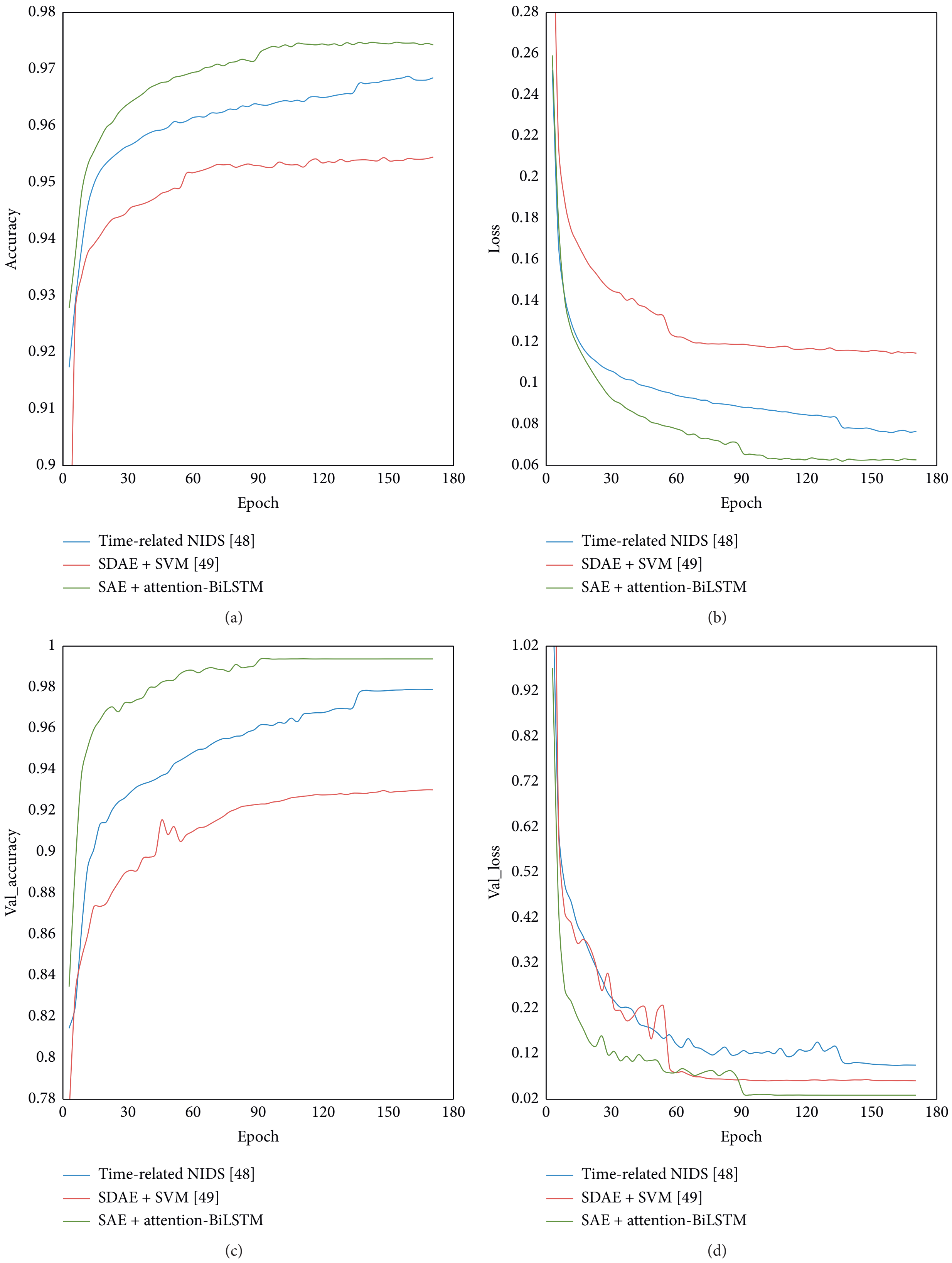
5.5. Comparison with Different Structures of SAE and LSTM. To explore the influence of different structures of SAE and LSTM on our proposed scheme, experiments were carried out with different SAE structures and LSTM structures. The SAE network uses three different structures: {128, 64, 32, 32}, {128, 64, 32}, and {128, 32, 32}. LSTM uses two separate structures: the original model and the improved model. Table 9 highlights the different experimental performance results. When the Stacked Autoencoder structure adopts {128, 64, 32, 32} four hidden layer structures and the improved LSTM network in our scheme is employed to classification, the model has the best effect. But, the structure of a Stacked Autoencoder with four hidden layers will increase the convergence time of the algorithm. Improving LSTM to BiLSTM and adding the Attention Mechanism will also increase the amount of calculation in the model. Therefore, in the actual scenes in Smart Grid, it is necessary to comprehensively consider the detection accuracy, the required calculation configuration of the model, and the implementation time of the scheme. We should select the optimal structure for communication intrusion detection in AMI.

5.6. Comparison with Different Timesteps. Because our scheme uses the method of generating time series data in the final classification model, the selection of different timestep values will also affect the performance metrics of the model. In the LSTM classification model, timestep refers to how much data the current input data of the model is related to before. In the final classification process, through code debugging, the timestep value is selected in the set {4, 8, 12, 16, 20}, and we record the results of our model. Figure 12 shows the comparison of four performance metrics after different timesteps. With the increase of timesteps, the performance of various metrics gradually becomes better and eventually tends to a stable value. *Accuracy*, *Precision*, and *FAR* have the best performance when the timestep value is equal to 20. *Recall* has the best performance when the timestep value is equal to 16, but the *Recall* value is only 0.0005 lower when the timestep value is equal to 20. However, while improving the performance of the model, the calculation time of the model should be taken into account. Figure 13 explains the calculation time of each Epoch for different timesteps. It indicates that the cost of increasing timesteps to obtain good performance is that the calculation time becomes longer.

We set different timestep values in the classification model. Figure 14 records the accuracy and loss values of the training dataset and the testing dataset after each Epoch. As the timestep value increases, the final performance that the model can eventually achieve becomes better and better, but this does not mean that the time to reach the best performance is getting faster. Figure 14 suggests that the overall convergence trend is almost the same in the process of the model reaching the best performance. When the timestep value increases evenly, the performance improvement of the model is less and less obvious.

5.7. Experiments on the NSL-KDD Dataset. The proposed scheme shows excellent performance on the UNSW-NB15 dataset. To ensure the robustness of the proposed scheme, we use the NSL-KDD dataset for experiments. The NSL-KDD dataset is improved based on the KDD Cup99 dataset [50]. Compared to KDD Cup99, NSL-KDD does not include redundant records in the training dataset and duplicate records in the testing dataset. This dataset also conforms to the essential characteristics of traffic in the AMI communication environment: data imbalance and multidimensional feature structure. In the NSL-KDD dataset, there are 125973 records in the training dataset and 22544 records in the testing dataset. Table 10 shows the distribution of various types of intrusion attacks. In addition to the data marked as Normal, there are samples of four types of intrusion attacks: DoS, U2R, R2L, and Probe. The dataset has 42 features (1 category feature, 7 discrete features, and 34 continuous features).

Under the same environmental setting as 5.1, we conducted experiments on the NSL-KDD dataset according to the proposed scheme. Table 11 is the confusion matrix obtained through the testing dataset. Through the confusion matrix, we calculate the performance metrics, as shown in Table 12. The proposed scheme is also applicable to the NSL-KDD dataset, showing excellent performance in intrusion attack detection. Unlike the UNSW-NB15 dataset, the testing dataset in NSL-KDD includes many new attack variants. Therefore, the scheme could detect new attack variants.



(a)

(b)

(c)

(d)

Figure 10: Changes in accuracy and loss of the training dataset and the testing dataset with the increase of Epoch. (a) Accuracy changes in the training dataset. (b) Loss changes in the training dataset. (c) Accuracy changes in the testing dataset. (d) Loss changes in the testing dataset.

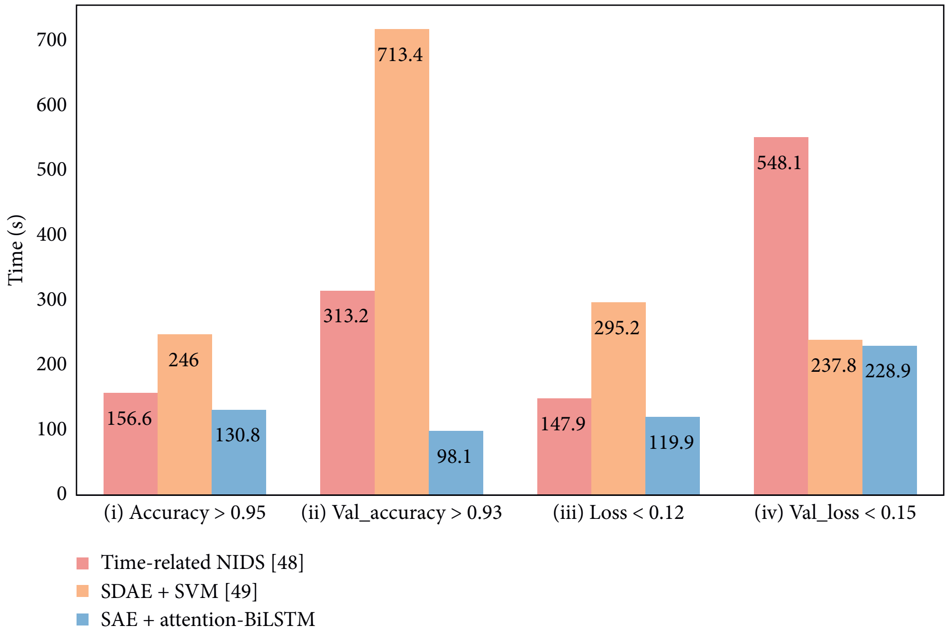


FIGURE 11: The running time of each program when the set threshold is reached. (i) The accuracy of the training dataset reaches 0.95. (ii) The accuracy of the testing dataset reaches 0.93. (iii) The loss of the training dataset reaches 0.12. (iv) The loss of the testing dataset reaches 0.15.

TABLE 6: Results of each performance evaluation metric on the UNSW-NB15 dataset.

Evaluation index	Value
Accuracy	0.9941
Precision	0.9914
Recall	0.9952
F1_score	0.9933
FAR	0.0069

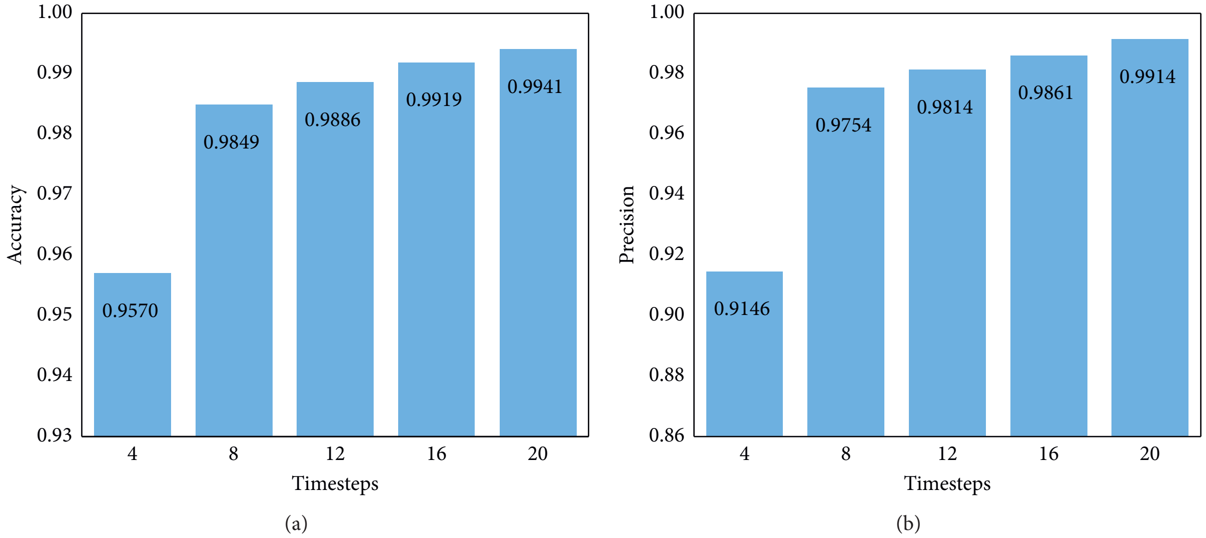
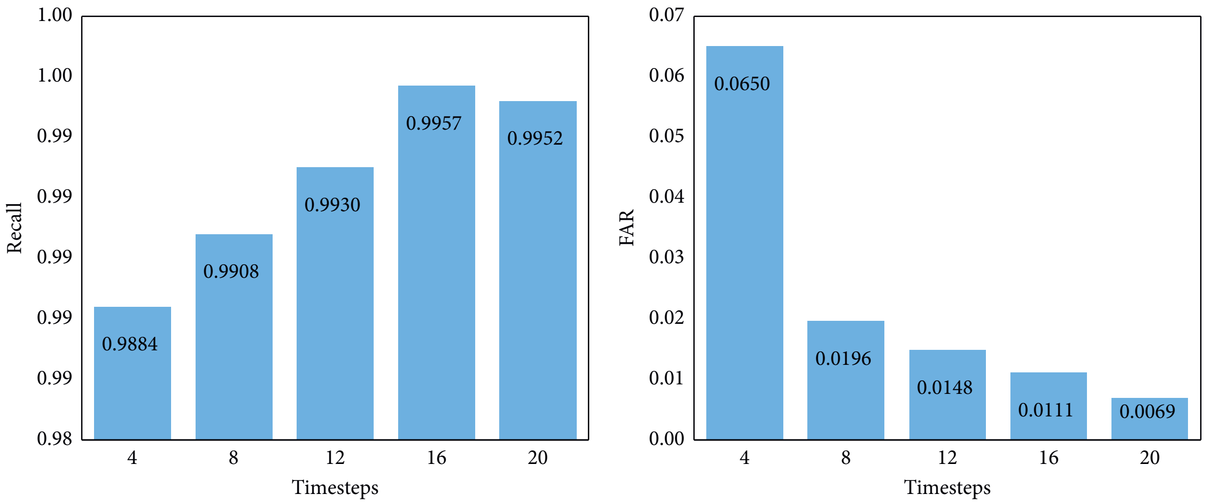


FIGURE 12: Continued.



(c) (d)

Figure 12: The results of performance metrics with different timesteps. (a) Accuracy. (b) Precision. (c) Recall. (d) FAR.

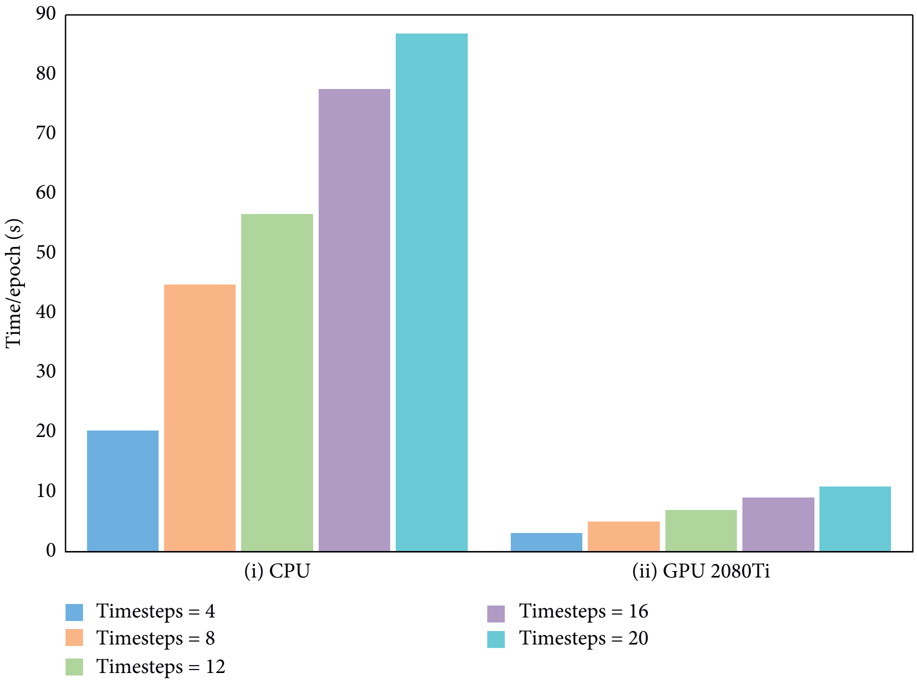


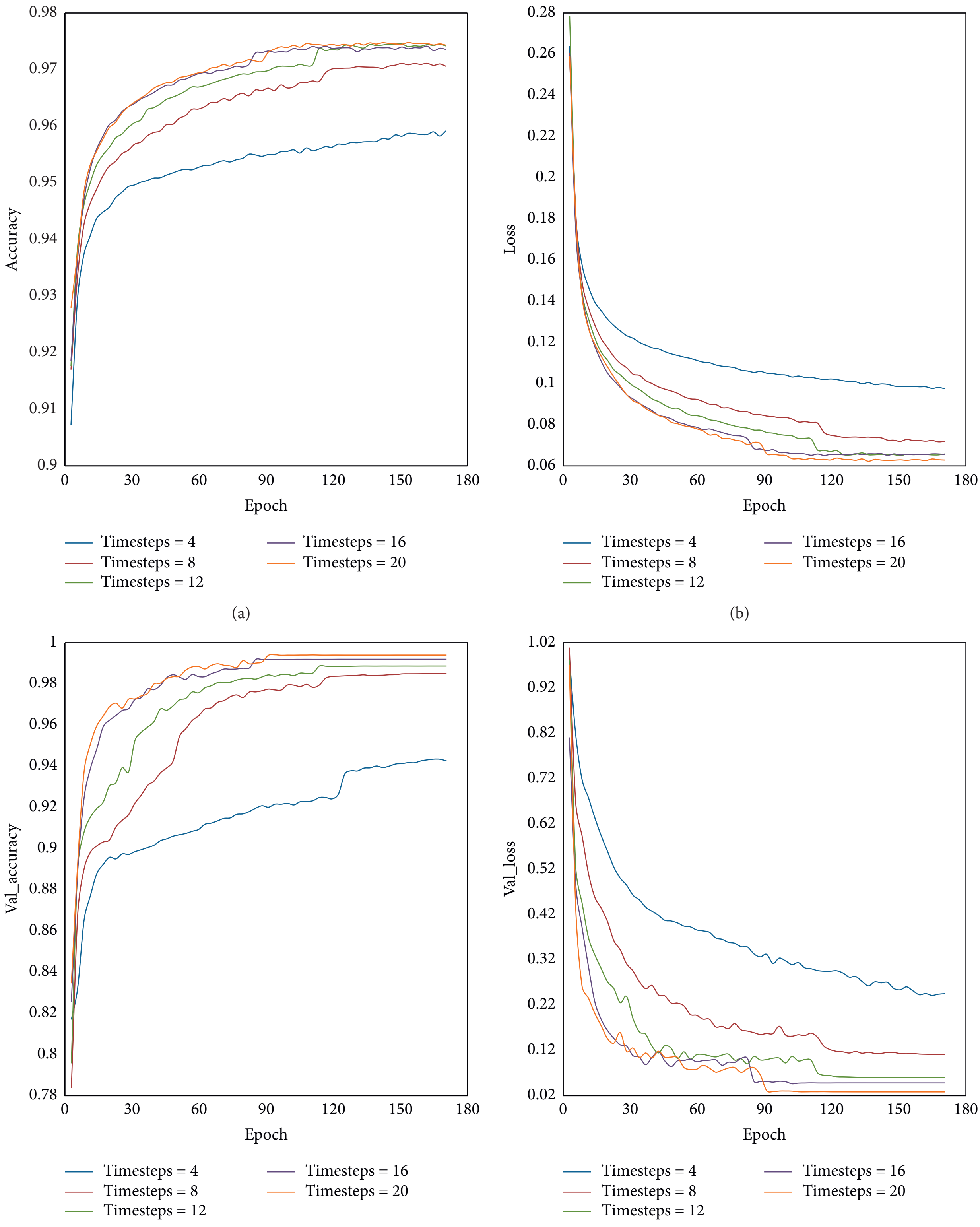
Figure 13: The calculation time of each Epoch under different timestep values. (i) The machine configuration used is CPU. (ii) The machine configuration used is GPU 2080Ti.

6. Threats to Validity

In this section, we report possible threats to the validity of the proposed scheme.

6.1. Threats to Internal Validity. In the process of intrusion detection, the dependent variable is the performance evaluation metric, which is obtained finally. It is calculated based on the confusion matrix of the classification results. Time performance is also used as one of the evaluation metrics in comparison with related work. However, we should ensure the consistency of the machines used when comparing the time performance of schemes. GPU will significantly accelerate the training speed of the model.

The independent variables that affect the dependent variable can be split into the structural setting of the model and the internal parameters of the model. In the structure of



(a)

(b)

(c)

(d)

Figure 14: Changes in performance metrics with increasing Epoch when setting different timesteps. (a) Accuracy changes in the training dataset. (b) Loss changes in the training dataset. (c) Accuracy changes in the testing dataset. (d) Loss changes in the testing dataset.

Table 7: The comparison of results between the proposed method and traditional machine learning methods.

Method	Accuracy	Precision	Recall	FAR
K-nearest neighbor	0.7544	0.7942	0.6977	0.1870
Naive Bayesian	0.8358	0.7399	0.8731	0.1869
Decision tree	0.8531	0.7811	0.8765	0.1624
Random forest	0.8583	0.8434	0.8400	0.1268
Logistic regression	0.8471	0.7212	0.9190	0.1917
Support vector machines	0.6486	**0.9964**	0.5599	0.0079
Multilayer perceptron	0.8095	0.7147	0.8350	0.2063
SAE + Attention-BiLSTM	**0.9941**	0.9914	**0.9952**	**0.0069**

Table 8: The comparison of results among the proposed scheme and recent intrusion detection papers.

Method	Accuracy	Precision	Recall	FAR
DO_IDS [47]	0.9282	0.9670	0.8966	0.0330
Time-related NIDS [48]	0.9793	0.9685	0.9848	0.0251
SDAE + SVM [49]	0.9311	0.9679	0.8879	0.0280
SAE + Attention-BiLSTM	0.9941	0.9914	0.9952	0.0069

Table 9: Evaluation results of SAE and LSTM with different structures.

Structures of SAE and LSTM		Accuracy	Precision	Recall	FAR
{128, 64, 32, 32}	Attention-BiLSTM	**0.9941**	**0.9914**	**0.9952**	**0.0069**
	LSTM	0.9859	0.9778	0.9905	0.0178
{128, 64, 32}	Attention-BiLSTM	0.9893	0.9822	0.9938	0.0142
	LSTM	0.9809	0.9682	0.9888	0.0252
{128, 32, 32}	Attention-BiLSTM	0.9861	0.9763	0.9924	0.0189
	LSTM	0.9767	0.9595	0.9880	0.0320

Table 10: The distribution of various types of intrusion attacks in the NSL-KDD dataset.

Dataset class	Attack class	Size	Distribution (%)
Training dataset	Normal	67343	53.46
	DoS	45927	36.46
	Probe	11656	9.25
	R2L	995	0.79
	U2R	52	0.04
Testing dataset	Normal	9711	43.07
	DoS	7458	33.08
	Probe	2421	10.74
	R2L	2754	12.22
	U2R	200	0.89

Table 11: Confusion matrix over the NSL-KDD dataset using SAE + Attention – BiLSTM.

		Predicted	
		Anomalous	Normal
Actual	Anomalous	11769	95
	Normal	46	12826

Table 12: Results of each performance evaluation metric on the NSL-KDD dataset.

Evaluation index	Value
Accuracy	0.9943
Precision	0.9961
Recall	0.9920
F1_score	0.9940
FAR	0.0036

the model, the number of layers in the SAE and the number of neurons in each layer will have a significant impact on the internal validity. When using LSTM for classification, choosing a bidirectional structure to extract features and adding an Attention Mechanism will improve accuracy. However, the complexity of the model structure will lead to an increase in the computational cost. Timesteps is one of the most important internal parameters, which affects the time performance of the model. In the programming models for deep learning, we also need to consider the threat of internal parameters (such as learning rate) to internal validity. Besides, some advanced technologies are used in the algorithm design of our scheme. For example, we use the *dropout* function to prevent the model from overfitting. Different probabilities of deactivation may cause the model to produce different classification results.

Table 13: Attack categories and subtypes of attacks in the NSL-KDD dataset.

Categories of attack	Subtype
DoS	apache2, back, land, mailbomb, neptune, pod, processtable, smurf, teardrop, udpstorm, worm
U2R	buffer_overflow, loadmodule, perl, ps, sqlattack, rootkit, xterm
R2L	ftp_write, guess_passwd, httptunnel, phf, imap, multihop, named, sendmail, snmpgetattack, snmpguess, spy, warezclient, warezmaster, xlock, xsnoop
Probe	ipsweep, portsweep, mscan, nmap, saint, satan

6.2. Threats to External Validity. Aiming at the characteristics of intrusion attack traffic in AMI, we propose a detection scheme. In the experiment, the UNSW-NB15 dataset was used to verify the effectiveness of the scheme. To ensure the robustness of the scheme, we use the NSL-KDD dataset to test. These two general datasets are collected by network communication, so the proposed scheme can adapt to other environments. In other industrial control traffic anomaly detection, there are also problems with high-dimensional features and data imbalance. We implement feature dimensionality reduction through the SAE part of the designed IDS and then use BiLSTM with Attention Mechanism to extract the bidirectional feature structure. Finally, the purpose of efficient intrusion detection can be achieved.

But in deep research, we found that the proposed scheme needs the ability to detect subtypes of attacks. For example, Table 13 shows the attack categories and subtypes of attacks in the NSL-KDD dataset. Based on the results of subtype intrusion attack detection, researchers can implement defense measures precisely.

7. Conclusion

In this paper, considering high-dimensional features of massive data and data imbalance in AMI, we propose the corresponding intrusion detection scheme. The scheme consists of two parts: feature dimensionality reduction and classification. In feature dimensionality reduction, we use the Stacked Autoencoder to convert the 196-dimensional original data features into 32-dimensional encoded data features. It could reduce the computational complexity of the model. In classification, the Attention Mechanism is applied to the BiLSTM model to determine the criticality of the dimensionality and select efficient features to improve the accuracy of classification. The proposed intrusion detection scheme is evaluated using the UNSW-NB15 dataset. Through experimental comparison: among all the performance indicators selected in this paper, the proposed communication intrusion detection scheme in AMI is much better than the methods based on traditional machine learning. In comparison with the new intrusion detection scheme, our scheme still shows good performance. By changing the structure of the SAE and LSTM models in the scheme and debugging the timestep value, we explore the influence of the internal parameters of the model on the overall scheme. To ensure the robustness of the work, we test on the NSL-KDD dataset. It also shows superior performance.

In future work, although the accuracy of intrusion detection is improved and the FAR is reduced, the scheme we propose has a lot of time cost in terms of the SAE deep network structure and the calculation of the Attention Mechanism. Therefore, it is necessary to further explore how to reduce the computational cost of the model while ensuring high accuracy and low FAR. In the communication scenario of AMI, various new types of attacks emerge endlessly. In our future work, it is crucial to choose the structure and corresponding parameters to realize the optimized intrusion detection scheme and adapt to the changes of new attacks in AMI.

Data Availability

The UNSW-NB15 dataset and the NSL-KDD dataset used to support the findings of this study are included within the paper. Besides, other data 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 by the National Natural Science Foundation of China (Grant No. 61772327), State Grid Gansu Electric Power Company Electric Power Research Institute (Grant No. H2019-275), and Shanghai Engineering Research Center on Big Data Management System (Grant No. H2020-216).

References

[1] M. A. Faisal, Z. Aung, J. R. Williams, and A. Sanchez, "Data-stream-based intrusion detection system for advanced metering infrastructure in smart grid: a feasibility study," *IEEE Systems Journal*, vol. 9, no. 1, pp. 31–44, 2015.

[2] F. M. Cleveland, "Cyber Security Issues for Advanced Metering Infrasttructure (AMI)," in *Proceedings of the 2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy In the 21st Century*, Pittsburgh, PA, USA, July 2008.

[3] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, "Securing the smart grid: a comprehensive compilation of intrusion detection and prevention systems," *IEEE Access*, vol. 7, pp. 46595–46620, 2019.

[4] A. Thakkar and R. Lohiya, "A review of the advancement in intrusion detection datasets," *Procedia Computer Science*, vol. 167, pp. 636–645, 2020.

[5] S. Han, M. Xie, H. Chen, and Y. Ling, "Intrusion detection in cyber-physical systems: techniques and challenges," *IEEE Systems Journal*, vol. 8, no. 4, pp. 1052–1062, 2014.

[6] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule based intrusion detection

system: analysis on UNSW-NB15 data set and the real time online dataset," *Cluster Computing*, vol. 23, no. 2, pp. 1397–1418, 2019.

[7] Z. Wang, S. Xu, G. Xu et al., "Game theoretical method for anomaly-based intrusion detection," *Security and Communication Networks*, vol. 2020, Article ID 8824163, 10 pages, 2020.

[8] C. Gu, S. Zhang, and H. Lu, "Online Internet intrusion detection based on flow statistical characteristics," *Knowledge Science, Engineering and Management*, vol. 7091, pp. 160–170, 2011.

[9] T. Mehmood and H. B. M. Rais, "Machine learning algorithms in context of intrusion detection," in *Proceedings of the 2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*, Kuala Lumpur, Malaysia, August 2016.

[10] N. Altwaijry, A. ALQahtani, and I. AlTuraiki, "A deep learning approach for anomaly-based network intrusion detection," *Big Data and Security*, vol. 1210, pp. 603–615, 2020.

[11] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, "A survey of deep learning-based network anomaly detection," *Cluster Computing*, vol. 22, no. S1, pp. 949–961, 2017.

[12] Y. Dong, R. Wang, and J. He, "Real-time network intrusion detection system based on deep learning," in *Proceedings of the 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, China, October 2019.

[13] S. A. Althubiti, E. M. Jones, and K. Roy, "LSTM for anomaly-based network intrusion detection," in *Proceedings of the 2018 28th International Telecommunication Networks And Applications Conference (ITNAC)*, Sydney, Australia, November 2018.

[14] D. M. Farid, H. Nouria, and M. Z. Rahman, "Combining naive bayes and decision tree for adaptive intrusion detection," *International Journal of Network Security & Its Applications*, vol. 2, no. 2, pp. 12–25, 2010.

[15] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, "An anomaly-based intrusion detection system for the smart grid based on cart decision tree," in *Proceedings of the 2018 Global Information Infrastructure and Networking Symposium (GIIS)*, Thessaloniki, Greece, October 2018.

[16] B. Senthilnayaki, K. Venkatalakshmi, and A. Kannan, "Intrusion detection system using fuzzy Rough set feature selection and modified KNN classifier," *International Arab Journal of Information Technology*, vol. 16, no. 4, pp. 746–753, 2019.

[17] R. Vijayanand, D. Devaraj, and B. Kannapiran, "Support vector machine based intrusion detection system with reduced input features for advanced metering infrastructure of smart grid," in *Proceedings of the 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, January 2017.

[18] T. Hurley, J. E. Perdomo, and A. Perez-Pons, "Hmm-based intrusion detection system for software defined networking," in *Proceedings of the 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Anaheim, CA, USA, December 2016.

[19] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications*, vol. 67, pp. 296–303, 2017.

[20] B. Ismail, R. In-Ho, and S. Ravi, "An intrusion detection system based on multi-level clustering for hierarchical wireless sensor networks," *Sensors*, vol. 15, no. 11, pp. 28960–28978, 2015.

[21] F. Salo, A. B. Nassif, and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," *Computer Networks*, vol. 148, pp. 164–175, 2019.

[22] J. Sun, X. Wang, N. Xiong, and J. Shao, "Learning sparse representation with variational auto-encoder for anomaly detection," *IEEE Access*, vol. 6, pp. 33353–33361, 2018.

[23] S. Ali and Y. Li, "Learning multilevel auto-encoders for DDoS attack detection in smart grid network," *IEEE Access*, vol. 7, pp. 108647–108659, 2019.

[24] A. Bhardwaj, V. Mangat, and R. Vig, "Hyperband Tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in cloud," *IEEE Access*, vol. 8, pp. 181916–181929, 2020.

[25] J. Gao, L. Gan, F. Buschendorf et al., "LSTM for SCADA intrusion detection,," in *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Victoria, Canada, August 2019.

[26] A. F. M. Agarap, "A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data," in *Proceedings of the 2018 10th International Conference On Machine Learning And Computing (ICMLC 2018)*, Macau, China, February 2018.

[27] B. Roy and H. Cheung, "A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network," in *Proceedings of the 2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, Sydney, Australia, November 2017.

[28] M. Khan, M. Karim, and Y. Kim, "A scalable and hybrid intrusion detection system based on the convolutional-LSTM network," *Symmetry*, vol. 11, no. 4, p. 583, 2019.

[29] B. Riyaz and S. Ganapathy, "A deep learning approach for effective intrusion detection in wireless networks using CNN," *Soft Computing*, vol. 24, no. 22, pp. 17265–17278, 2020.

[30] Z. Lin, Y. Shi, and Z. Xue, "Generative adversarial networks for attack generation against intrusion detection," *CoRR Abs*, vol. 2, no. 1, pp. 12–18, 2018.

[31] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks, in Advances in neural information processing systems 19," in *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, December 2006.

[32] H. Ma, S. Ma, Y. Xu et al., "Image denoising based on improved stacked sparse denoising autoencoder," *Computer Engineering and Applications*, vol. 54, no. 04, pp. 199–204, 2018.

[33] B. Abolhasanzadeh, "Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features," in *Proceedings of the 2015 7th Conference On Information And Knowledge Technology (IKT)*, Urmia, Iran, May 2015.

[34] X. Wang and X. Lu, "A host-based anomaly detection framework using XGBoost and LSTM for IoT devices," *Security and Communication Networks*, vol. 2020, Article ID 8838571, 13 pages, 2020.

[35] R. Hwang, M. Peng, V. Nguyen, and Y. Chang, "An LSTM-based deep learning approach for classifying malicious Traffic at the packet level," *Applied Sciences*, vol. 9, no. 16, p. 3414, 2019.

[36] E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*, Apress, New York, NY, USA, 2019.

[37] J. Zabalza, J. Ren, J. Zheng et al., "Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging," *Neurocomputing*, vol. 185, pp. 1–10, 2016.

[38] V. Singhal, A. Gogna, and A. Majumdar, "Deep dictionary learning vs deep belief network vs stacked autoencoder: an empirical analysis," in *Proceedings of the International Conference on Neural Information Processing*, vol. 9950, Springer International Publishing, Bangkok, Thailand, November 2016.

[39] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," 2015, https://arxiv.org/abs/1502.03167.

[40] W. Chen, S. Yang, X. A. Wang, W. Zhang, and J. Zhang, "Network malicious behavior detection using bidirectional LSTM," *Advances in Intelligent Systems and Computing*, vol. 772, pp. 627–635, 2018.

[41] N. Xianyang, C. Yinbao, and W. Zhongyu, "Remote sensing semantic segmentation with convolution neural network using attention mechanism," in *Proceedings of the 2019 14th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, Changsha, China, November 2019.

[42] L. Shen, B. Zou, Y. Hong et al., "The emerging enernet: convergence of the smart grid with the Internet of Things," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, November 2017.

[43] Y.-L. Li, S. Wang, and HAR-Net, "HAR-net: joint learning of hybrid attention for single-stage object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 3092–3103, 2020.

[44] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1-3, pp. 18–31, 2016.

[45] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive dataset for network intrusion detection systems (UNSW-NB15 network dataset)," in *Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, Australia, November 2015.

[46] A. D. Forbes, "Classification-algorithm evaluation: five performance measures based onconfusion matrices," *Journal of Clinical Monitoring*, vol. 11, no. 3, pp. 189–206, 1995.

[47] J. Ren, J. Guo, W. Qian et al., "Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms," *Security and Communication Networks*, vol. 2019, Article ID 7130868, 11 pages, 2019.

[48] Y. Lin, J. Wang, Y. Tu, L. Chen, and Z. Dou, "Time-related network intrusion detection model: a deep learning method," in *Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, USA, December 2019.

[49] W. Wang, X. Du, D. Shan, and N. Wang, "A hybrid cloud intrusion detection method based on SDAE and SVM," in *Proceedings of the 12th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, Xiangtan, China, October 2019.

[50] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, Canada, July 2009.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6661954, 8 pages
https://doi.org/10.1155/2021/6661954

Research Article

Machine Learning-Based Stealing Attack of the Temperature Monitoring System for the Energy Internet of Things

Qiong Li,[1] Liqiang Zhang,[2] Rui Zhou,[2] Yaowen Xia,[1] Wenfeng Gao ,[1] and Yonghang Tai [2]

[1]*Solar Energy Research Institute, Yunnan Normal University, Kunming, Yunnan 650500, China*
[2]*Yunnan Key Laboratory of Opto-electronic Information Technology, Yunnan Normal University, Kunming 650000, China*

Correspondence should be addressed to Wenfeng Gao; 413900096@qq.com and Yonghang Tai; taiyonghang@126.com

Received 18 December 2020; Revised 31 January 2021; Accepted 9 March 2021; Published 25 March 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Qiong Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the development of the Energy Internet of Things (EIoT), it is of great practical significance to study the security strategy and intelligent control system for solar thermal utilization system to optimize the operation efficiency and carry out intelligent dynamic adjustment. For buildings integrated with solar water heating systems, computational fluid dynamics simulation was used in analyzing the process of solar energy output. A method based on machine learning is proposed to predict energy conversion. Besides, the simulation and analysis are carried out in combination with the possible safety problems such as the vibration of the control system. This paper proposed a novel platform of EIoT for machine learning-based cybersecurity study and implemented the platform for the temperature monitoring system. After the evaluation of the machine learning-based cybersecurity study, the EIoT system demonstrated a high performance with the Extreme Gradient Boosting (XGBoost) training algorithm.

1. Introduction

With the resonant coupling of multiple energy sources, the new energy supply system under the background of the Internet of Things will have the characteristics of multi-energy complementation and synergy [1–3]. The traditional thermal and electric demand response will gradually develop into a comprehensive demand response suitable for integrated energy systems. The EIoT (Energy Internet of Things) has brought a better operation, monitoring, and management mode for the new energy utilization systems. It uses advanced sensors, control, and software applications to connect a large number of equipment, machines, and systems at the energy production side, energy transmission side, and energy consumption side to form the "Internet of Things foundation" [4]. Big data analysis, machine learning, and prediction are the important technical support for the EIoT to realize the life characteristics. Therefore, in the solar water heating system, the sensor measurement technology, numerical simulation technology, Internet of Things technology, and machine learning technology can be used to remote monitoring of the control parameters, heat gain, and status of the solar water heating engineering system, so as to realize the saving, comfortable, efficient and reliable water, energy and heat consumption, which has strong practical application value [5].

However, in the operation environment of EIoT, the operation control of a high-proportion new energy system highly relies on low-latency and highly reliable information and communication technology, which brings excellent technical challenges to the system's network security defense.

In the applications of IoT, in recent years, some critical solutions have been put forward by many experts and scholars to security issues. Xu [6] listed some security problems and critical technologies of IoT. Mahmoud et al. [7] provide eight frameworks for the security applications in the IoT field, each of the framework addresses in detail the security measures and the ability to fight against the attacks. In each specific application scenario, faced with the security demand of the smart home environment, Huichen Lin and Neil Bergmann [8] proposed two critical technologies of auxiliary management to deal with these problems. Minoni et al. [9] provided relevant tools and technologies to address security vulnerabilities in e-health and assisted living applications. Baranwal et al. [10] designed a remote control and monitoring device; it can be used for the security detection of farmland, grain storage, and cold storage. At the same time, ML can provide algorithm support for the designed whole IoT system. For example, when we develop a system that uses text data for classification, we can adopt the ML method to process the data and set our system according to the actual situation. The researchers above have given us an idea to use ML to design a new platform for the temperature monitoring system. The system is equipped with high security and privacy and can be applied to daily life.

(1) Create a new platform of EIoT for machine learning-based cybersecurity study

(2) Propose a model stealing attack on the intelligent energy supply system

(3) Implement the proposed intelligent energy supply platform and model stealing attack

The structure and content of this paper are organized as follows: in Section 2, we review the related works on the cyber-attacks with machine learning for the EIoT. The model stealing attack experiments are designed in the methodology part, which is presented in Section 3. In the next section, the performance of attacks on the medical platform was demonstrated and discussed. In the last section, we summarize the results and conclude this paper.

2. Related Work

Another algorithm in machine learning, the RandomForest algorithm, also is used in systems of IoT. In the face of IoT security problems, Nawir et al. [11] directly summarized various attacks into a well-structured classification to help researchers and developers, so that security measures can be planned appropriately in the development of the IoT. Overall, with the extensive application in IoT of various fields, the security of IoT will always be a hot research direction. However, sometimes existing solutions are insufficient to cover the security range of IoT; machine learning (ML) technology can provide embedded intelligence in IoT devices and networks to address security issues [12]. According to the unique characteristics of IoT devices and environments, Zeadally et al. [13] mentioned the relevant advantages of ML algorithms. Gomes et al. [14] used this set of an algorithm for indoor positioning of users in a smart home. As a basic algorithm in machine learning, the XGBoost algorithm is applied to a wide range of modern industries. The main advantage of XGBoost is its scalability, and speed of execution is usually superior to other ML models [15]. When dealing with classification problems, compared to other models, XGBoost has better classification accuracy [16]. In the field of IoT, sometimes XGBoost is used as a method to detect if a system has been compromised [17]. In this paper, XGBoost has been applied to solve our data classification; on this basis, our system is also equipped with high security.

3. New Platform for Mobile and Intelligent Medicine

3.1. EIoT System Design

3.1.1. Machine Learning Models for Monitoring Water Instantaneous Flow. In this article, we use RandomForest to classify the water temperature measured from the water tank outlet and then use XGBoost to steal the entire network. RandomForest is a classifier containing multiple decision trees, and its output category is determined by the model number of the categories output by individual trees. It was first proposed by Leo Breiman and Adele Cutler. We can think of a decision tree as a collection of if-then rules. Decision tree learning can be described by $Pi = Xi/M,\ i = 1, 2, \ldots, N$, in which x is the input instance (eigenvector), M is the number of features, and i is the class tag, $i = 1, 2, \ldots, N$. N is the sample size [breiman] RF constructs bagging integration on a decision tree-based learner and further introduces random attribute selection in the training process of the decision tree. XGBoost is a tree integration model. Assuming that there are k trees, so the sum of the predicted values of each of the k trees for the sample is used as the prediction of the XGBoost model.

Given a dataset, including z samples and s features, $\mathscr{T} = \{(x_i, y_i)\}$ $(|\mathscr{T}| = z, x_i \epsilon \mathscr{R}^s, y_i \epsilon \mathscr{R})$.

The output of the tree model is

$$\hat{y}_i = \varnothing(x_i) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \varepsilon \mathscr{F}. \tag{1}$$

The space of CART tree is F, as follows:

$$\mathscr{F} = \{f(x) = \omega_{q(x)}\}\left(q\colon \mathscr{R}^s \longrightarrow H, \omega \varepsilon \mathscr{R}^H\right), \tag{2}$$

where q represents the model of the tree. Input a sample and map the selection to the leaf node according to the model to output the predicted score. $\omega_{q(x)}$ represents the set of fractions of all leaf nodes of tree Q; H is the number of leaves in the tree q. Therefore, it can be seen from equation (1) that the predicted value of XGBoost is the sum of the predicted values of each tree; namely, the sum of the scores of the corresponding leaf nodes of each tree (ω) i represents the score of the ith leaf node. Our goal is to learn K tree models like this $f(x)$. First, define a target function:

$$\mathscr{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \text{ where } \Omega(f) = \gamma T + \frac{1}{2}\lambda\omega^2. \tag{3}$$

The optimization parameter of the XGBoost model is model F (x), rather than an additive value, so we cannot use traditional optimization methods to optimize the Euclidean distance. We use additive training to learn the model.

3.2. Model Stealing Attack to the New Platform

3.2.1. Overview of the Threat Model. As we can see in Figure 1, the IoT temperature control system is divided into three different layers by us. In the player, we focus on collecting the functions that users need to implement in this system. In this article, it is precisely the temperature monitoring of the water tank outlet; the second layer is the *n* layer, focusing on modelling the entire user requirements. In this article, the RandomForest algorithm is mainly used to monitor the temperature of the water tank, and after experiencing our system, users will provide suggestions, and then we make improvements to our system based on these suggestions. This layer is to feed back user opinions to us so that our network can better meet customer needs.

Dividing the IoT network into three different layers is the result of consideration from the perspective of network security. Doing so can reduce the risk of network attacks on the entire system. In the perceptual layer, Ian et al. [18] proposed the concept of input, which forms input by imposing small but deliberate worst-case perturbations on the examples in the dataset, so that they can output a high-confidence wrong answer.

For the solar hot water monitoring platform, its main purpose is to monitor and effectively dispatch the relevant data in the solar hot water system, such as monitoring the water level data of the water tank, and setting different water replenishment strategies according to the change of water level; or setting different water temperature heating strategies, so that the solar hot water system with insufficient sunlight can heat up automatically in time. Therefore, the system needs to effectively receive the data from the data acquisition end and establish the corresponding database, so as to obtain the corresponding data change curve, such as the water temperature change curve of the water tank, so as to facilitate the subsequent prediction of the water temperature change and set the corresponding maintenance strategy.

3.2.2. Theoretical Description of the Model Stealing Attack. We will introduce how to use data stolen from an existing target network (RandomForest in this case) to build our copycat network, in this part. The importance of the whole process is to use random natural data to instruct a network of imitators from the existing target network. Two steps have been included: the creation of pseudo training data is used to train the imitator network. First, use the target network as a grey box to label random natural data to generate a pseudo dataset. Then, use the pseudo dataset to train the simulated network and copy the attributes of the target network (Figure 2).

In Figure 2, we briefly explained how to build a copycat network. Now, we will introduce this process in detail. Corresponding to Figure 2, the entire copycat network training process should be divided into two parts. The first is an essential training set. The training set used by copycat builds and the training set used by the target network are in the same problem [19–21]. The point that needs to be emphasized here is that although they use data in the same problem domain, their datasets are not the same, because random sampling is generally used when collecting data. If there is a large amount of data duplication when adopting the copycat dataset, we can also go online and use natural data to augment the copycat dataset, so as to avoid copycat and the target network using the same dataset and affecting the final result judgment error. Importing the selected dataset into the original professional network to steal the corresponding labels is the most critical step for the entire copycat network. At this time, the quality of the titles stolen will often determine whether the copycat network can fully implement the original network.

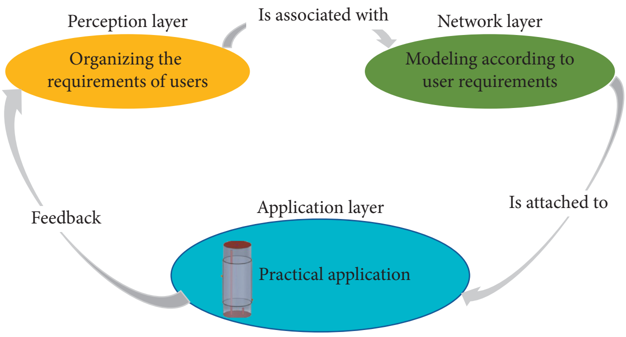


Figure 1: EIoT system framework based on the temperature control system.

After using the copycat dataset to steal the appropriate labels from the target network, the next step is to use these datasets to train the selected network. In this article, we chose the XGBoost *t* as the attack model. The reason for the choice is that it is the same machine learning algorithm as RandomForest, but there are also differences: RandomForest processes data in serial, while XGBoost processes data [22]. It is parallel processing. This choice of the network also proves the feasibility of our network attack from the side. During training, we imported the copycat dataset and its corresponding labels generated by ourselves into the XGBoost network and imported the same test set as the original network to determine whether the copycat network we created can achieve the accuracy of the average temperature and abnormal temperature in the input data classification.

Next, we will explain the assignability of adversarial samples. Suppose that the adversary is interested in classifying the wrong sample and producing a hostile sample $\overrightarrow{\omega^*}$ different from the model in which the class is assigned to the legal input $\vec{\omega}$. In the following optimization formula, we can achieve this:

$$\overrightarrow{\omega^*} = \vec{\omega} + \theta_{\vec{\omega}} \text{ where } \theta_{\vec{\omega}} = \arg\min_{\vec{\alpha}} g(\vec{\omega} + \vec{\alpha}) \neq g(\vec{\omega}). \tag{4}$$

$\overrightarrow{\omega^*}$ is the hostile sample, and g is the activation function. However, adversarial samples are often incorrectly classified as g' instead of g in practice. For the convenience of discussion, the concept of transferability of adversarial samples is formalized [23]:

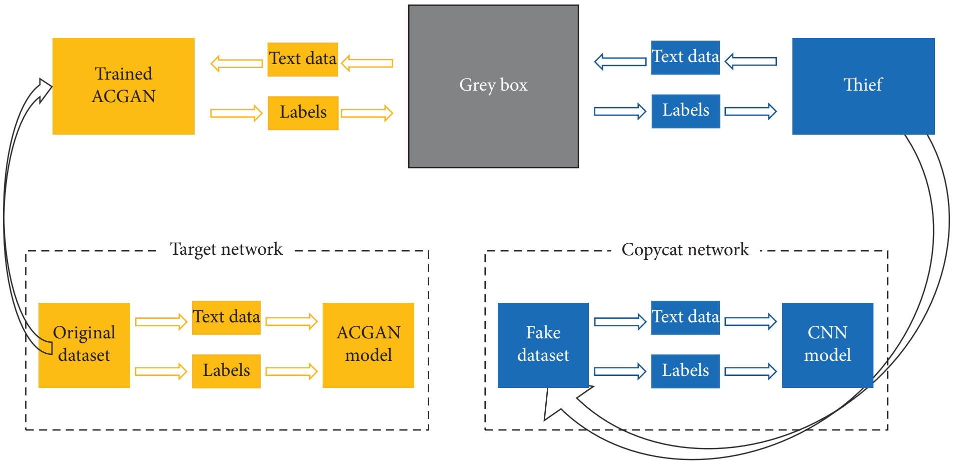


Figure 2: The figure shows how to steal a model from a trained network. The entire network can be divided into two parts; that is, copycat is divided into two parts. First of all, we see that the part on the left is the existing network, which is what we call the target network. It uses the data we already have for training and testing, while the network on the right is the copycat network. Its basics are that it is based on a certain cognition of the target network, and its training set and the training set used by the original network are in the same problem domain.

$$\Pi_Y(g, g') = \left|\left\{g'(\vec{\omega}) \neq g'\left(\vec{\omega} + \overrightarrow{\theta}_{\vec{\omega}} : \vec{\omega} \in Y\right)\right\}\right|. \quad (5)$$

Set Y represents expected input distribution solved by the models g and g' in the task. We divide the transferability of adversarial samples into two variables to describe the models (g, g'). The first is the transferability within the technology, the transferability between different parameter initializations of the same technology or training models of different datasets (for example, g and g' are both deep learning networks or both support vector machines (SVM)) is defined by which. Second, for cross-technology transferability, two technologies can be used to train models (for example, g is a deep learning network and g' is SVM).

4. Experiments and Results

4.1. Implementation of the Platform

4.1.1. Discussion on Specific Temperature Control System Usage Scenarios and the Attack. The solar heating system is the most widely used solar energy utilization system. The dynamic modelling, characteristic analysis, and optimal control of solar heating systems play an essential role in promoting intelligent applications, safety, and convenience. The numerical calculation method of the dynamic thermal characteristics of the solar heating system is an effective means for the thermal dynamic modelling and analysis of the heating pipe network. However, the time and space complexity of the numerical calculation method are relatively large, and it is necessary to deeply analyze the time and space of the numerical calculation method. Due to its excellent algorithm, machine learning can optimize the numerical calculation performance of the thermal dynamics of the heating system, thereby providing a basis for the rapid analysis and optimization of the non-steady thermal process of an extensive heating network. Moreover, when the heating network performs frequent and wide-range temperature and flows adjustments, it may cause the control system to oscillate, which in turn causes the dynamic instability and imbalance of the heating network, so that our temperature control system has its place.

The intelligent solar heating control system based on EIoT can be seen in Figure 3. Energy supply, storage and transfer, energy management system, and energy consumption have been contained within the system. The working process of the whole system can be described as follows: after the solar energy is captured by the energy acquisition system, the energy is converted into thermal energy for storage by heating cold water. After the energy monitoring system determines that the output water temperature reaches the safe water temperature, the energy will be output to the user's home.

4.1.2. Discussion on Specific Temperature Control System Usage Scenarios and the Attack. In this section, a solar water heating system for buildings is studied, and its hot water output process is simulated by computational fluid dynamics (CFD). Considering the actual operation of the solar balcony wall hanging system, the inlet temperature of the circulating working medium is set as 338 K, and the initial water temperature inside the water tank is 288 K. The temperature contours and velocity vector diagram change with time when the mass flow rate of the circulating working medium is 0.022 kg/s. The temperature and internal velocity fields change with time as shown in Figure 4. When the inlet cold water penetrates a certain height in the water tank, it falls back. The falling fluid sucks hot water from its adjacent hot water area, forcing the cooler fluid to move down to the bottom of the solar water tank gradually. At the same time, the hot water in the water tank is discharged through the tank outlet. However, with the increase of time, the temperature difference between the cold water and the hot water decreases.

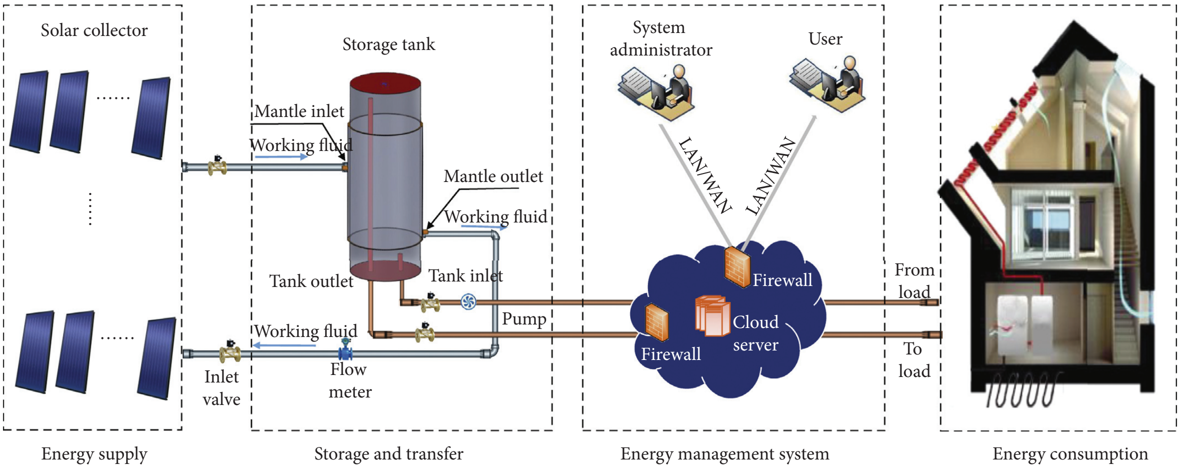


Figure 3: The intelligent solar heating control system based on EIoT.

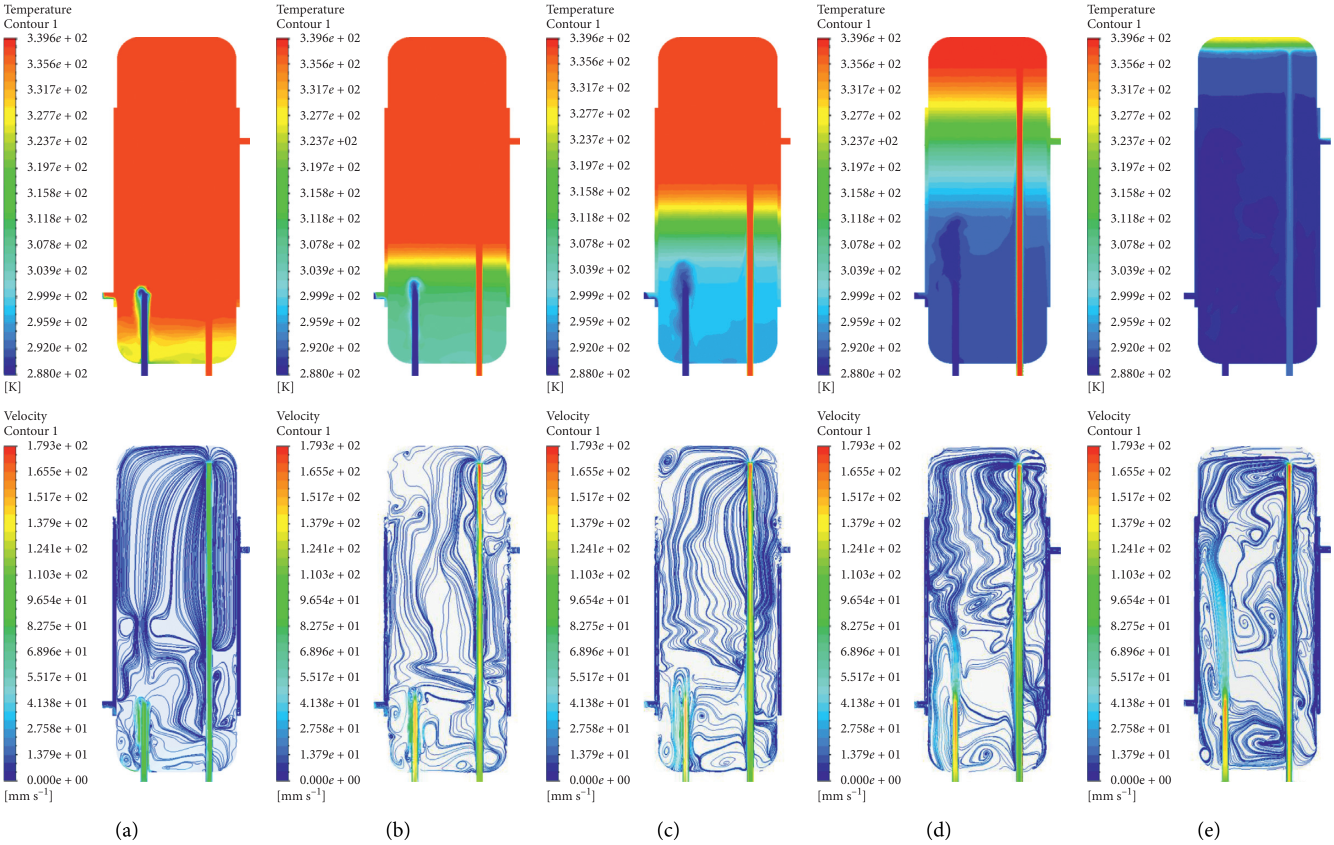


Figure 4: Temperature contours (top row) and velocity streamline (bottom row) in solar water storage tank in the discharging mode. (a) (t) = 100 s, (b) (t) = 900 s, (c) (t) = 1800 s, (d) (t) = 2700 s, and (e) (t) = 3600 s.

After we monitored the temperature of the water outlet of our experimental platform, we obtained a set of temperature records at different times under the same water flow conditions, including 909 sets of abnormal data and 230 sets of average data. Here, we classify the data based on the body's tolerance to water temperature during bathing: 310.15 (K) ~ 315.15 (K). All temperatures data within the range are classified as standard data, and data outside this temperature range are classified as abnormal data. The above is our design philosophy for the temperature monitoring system of the Internet of Things. However, if this temperature monitoring system is attacked by the network, the following results will be produced: the failure of the temperature alarm system will

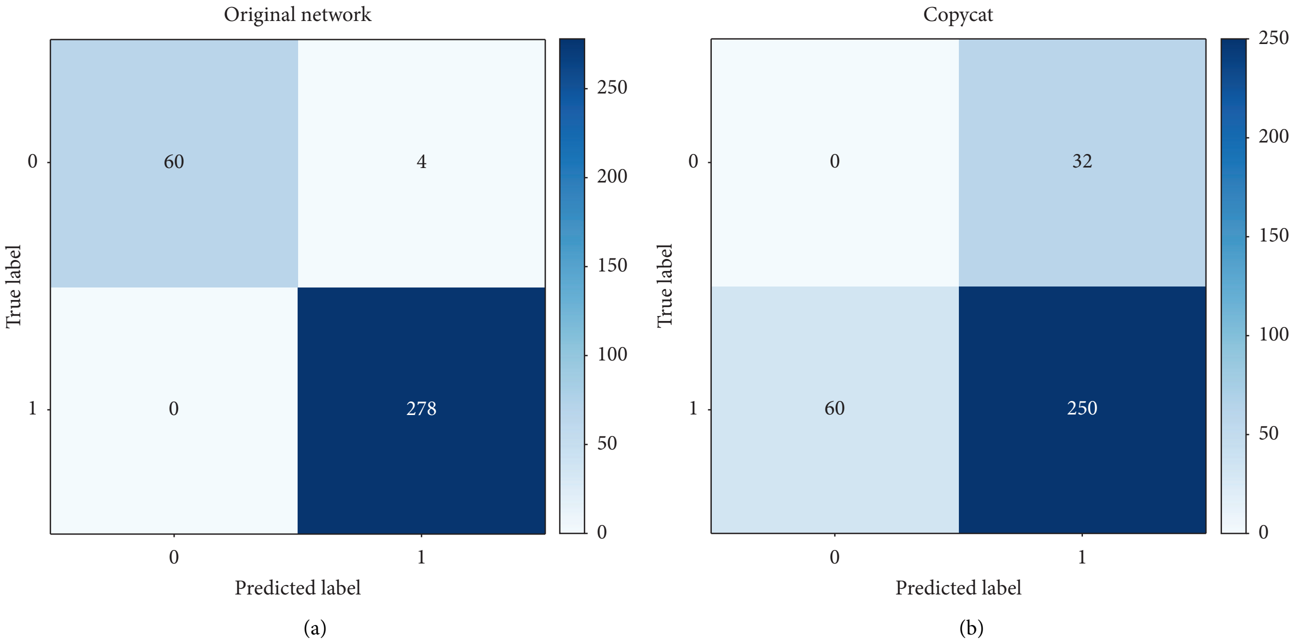


Figure 5: The confusion matrix for average temperature and abnormal temperature prediction.

threaten the personal safety of users. The theft of the entire network of the temperature control system will cause direct economic losses to the owner. And the whole copycat network we designed is based on stealing the whole of RandomForest prediction system to warn our system.

4.2. Performance of the Temperature Control System. The confusion matrix is generally used to evaluate the network output. The following is the definition of confusion matrix. Confounding matrix is a situation analysis table that summarizes the prediction results of classification models in machine learning. In the form of matrix, records in the dataset are summarized according to two criteria of real classification and classification judgment predicted by classification models. TP (True Positive), FN (False Negative), FP (False Positive), and TN (True Negative) are the four elements of the confusion matrix that reflect the performance of the model [24]. A high proportion of TP means that the performance of the entire network is satisfied.

In the predictive classification model, the larger the number of TP and TN, and the smaller the number of FP and FN, the higher the prediction accuracy (as can be seen from Figure 5). However, the calculations in the confusion matrix are numbers. Sometimes, in the face of large amounts of data, it is difficult to measure the number of models by counting. Therefore, the confusion matrix is an extension of the secondary and tertiary indicators (the lowest indicator addition, subtraction, multiplication, and division) in the basic statistical results [25].

Table 1: Values of different indicators based on the source model.

Object	Precision	Recall	F1-score
Normal T	0.94	1.00	0.97
Abnormal T	1.00	0.99	0.99
Macro avg	0.97	0.99	0.98
Weighted avg	0.99	0.99	0.99
Accuracy	—	—	0.99

$$\begin{aligned} Acc &= \frac{tp + tn}{tp + fp + fn + tn}, \\ \text{Rec} &= \frac{tp}{tp + tn}, \\ \text{pre} &= \frac{tp}{tp + tn}, \\ f_1 &= 2^* \frac{\text{Pre}^* \text{Rec}}{\text{Pre} + \text{Rec}}. \end{aligned} \tag{6}$$

We analyzed the confusion matrix obtained by RandomForest to classify the existing temperature dataset and found that TP and TN accounted for the highest proportion of the total output (278 + 60 = 338, the total of which is 342). Based on this data, we can draw a conclusion: using RandomForest to create a temperature control system can get a high accuracy.

Macro average refers to averaging the recall rates of category 1 and category 0. The weighted average is calculated using the proportion of the sample as the weight. It can be seen from the above table that our model has high prediction accuracy. As can be seen from Table 1, our model has reached a very high accuracy.

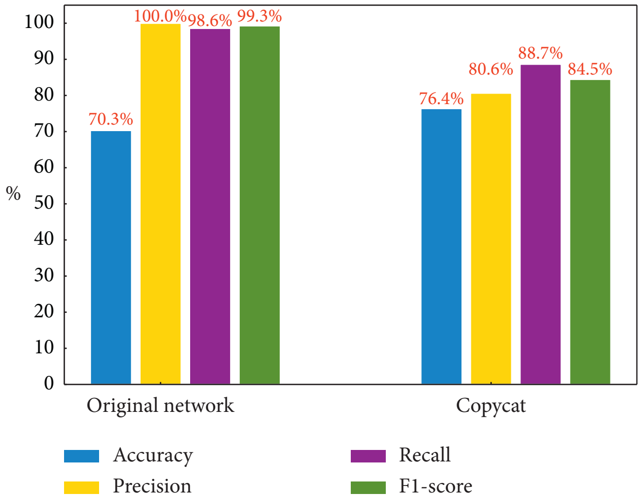


Figure 6: Comparison of different results between the original prediction model and copycat model.

Table 2: Values of different indicators based on a copycat model.

Object(copycat)	Precision	Recall	F1_score
Abnormal T	0.81	0.89	0.84
Macro avg	0.40	0.44	0.42
Weighted avg	0.66	0.73	0.70
Accuracy	—	—	0.73

4.3. Effectiveness of Model Stealing Attack. A RandomForest was trained to predict the average temperature and abnormal temperature. Based on the understanding of temperature data, we set the maximum depth of the (decision) tree in RandomForest to 30. In some other hyperparameters, min_samples_leaf is set to 3, min_samples_split is set to 4, n_estimators is set to 8000, and the values of verbose and n_jobs are both set to 1 [26]. The implementation is based on Pytorch and uses NVIDIA GTX 1070 GPU.

The difference between the original network and the copycat network is fully reflected in Figure 6. Although the precision, recall, and F1-score values obtained by the copycat network are about 15% lower than those of the original network, the accuracy obtained by the copycat network is 6.1% higher than that of the original network, which is enough to prove that the copycat network can pose a threat to the original network.

Based on the data in Table 2, we can conclude that the model that uses machine learning to classify text data can be stolen by different kinds of machine learning. It can be seen from Table 1 that, without considering the average temperature output, the model we used XGBoost to steal can be able to distinguish abnormal temperature data. So in terms of classifying abnormal temperature, the model we stole can already achieve this function.

5. Conclusions

A solar water temperature monitoring system based on the Internet of Things was established in this article. Based on the temperature monitoring system, we propose a RandomForest for normal temperature and abnormal temperature classification. In order to demonstrate the attack on the established model on the IoT platform, an XGBoost model was built by using a small number of labeled samples to steal the known target model. Experimental results show that the replication model can successfully replicate the performance of the target RandomForest, with small performance differences. The success of this attack shows that intellectual property rights such as artificial intelligence models similar to temperature monitoring systems that have been successfully established can be stolen. How to effectively solve these problems has become an urgent problem in the field of deep learning.

Data Availability

The data supporting the results of this study can be obtained from the corresponding author.

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 the National Natural Science Foundation of China (62062069, 62062070, and 62005235).

References

[1] K. Wang, Y. Wang, Y. Sun, S. Guo, and J. Wu, "Green industrial internet of things architecture: an energy-efficient perspective," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 48–54, 2016.

[2] Y. Kabalci, E. Kabalci, S. Padmanaban, J. B. Holm-Nielsen, and F. Blaabjerg, "Internet of things applications as energy internet in smart grids and smart environments," *Electronics*, vol. 8, no. 9, p. 972, 2019.

[3] S. O. Muhanji, A. E. Flint, and A. M. Farid, *Transactive Energy Applications of eIoT: The Development of the Energy Internet of Things in Energy Infrastructure*, eIoT, Berlin, Germany, 2019.

[4] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient energy management for the internet of things in smart cities," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 84–91, 2017.

[5] A. S. Sani, D. Yuan, J. Jin et al., "Cyber security framework for internet of things-based energy internet," *Future Generation Computer Systems*, vol. 93, pp. 849–859, 2018.

[6] X. Xiaohui, "Study on security problems and key technologies of the internet of things," in *Proceedings of the 2013 International Conference on Computational and Information Sciences*, pp. 407–410, IEEE, Hubei, China, June 2013.

[7] M. Ammar, G. Russello, and B. Crispo, "Internet of Things: a survey on the security of IoT frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018.

[8] H. Lin and N. Bergmann, "IoT privacy and security challenges for smart home environments," *Information*, vol. 7, no. 3, p. 44, 2016.

[9] D. Minoli, K. Sohraby, and B. Occhiogrosso, "Iot security (IoTsec) mechanisms for e-health and ambient assisted living

applications," in *Proceedings of the 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, pp. 13–18, IEEE, Philadelphia, PA, USA, August 2017.

[10] T. Baranwal and P. K. Pateriya, "Development of IoT based smart security and monitoring devices for agriculture. 2016 6th international conference-cloud system and big data engineering (confluence)," in *Proceedings of the 6th Conference on Cloud System and Big Data Engineering*, pp. 597–602, IEEE, Noida, India, January 2016.

[11] M. Nawir, A. Amir, N. Yaakob et al., "Internet of things (IoT): taxonomy of security attacks," in *Proceedings of the 2016, 3rd bInternational Conference on Electronic Design (ICED)*, IEEE, Phuket, Thailand, August 2016.

[12] F. Hussain, R. Hussain, S. A. Hassan et al., "Machine learning in IoT security: current solutions and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 99, 2020.

[13] S. Zeadally and M. Tsikerdekis, "Securing internet of things (IoT) with machine learning," *International Journal of Communication Systems*, vol. 33, no. 1, Article ID e4169, 2020.

[14] R. Gomes, M. Ahsan, and A. Denton, "Random forest classifier in SDN framework for user-based indoor localization," in *Proceedings of the 2018 IEEE International Conference on Electro/Information Technology (EIT)*, pp. 0537–0542, IEEE, Rochester, MI, USA, May 2018.

[15] M. Ruiz-Abellón, A. Gabaldón, and A. Guillamón, "Load forecasting for a campus university using ensemble methods based on regression trees," *Energies*, vol. 11, no. 8, p. 2038, 2018.

[16] N. Manju, B. S. Harish, and V. Prajwal, "Ensemble feature selection and classification of internet traffic using XGBoost classifier," *International Journal of Computer Network & Information Security*, vol. 11, no. 7, 2019.

[17] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for IoT applications," *Wireless Personal Communications*, vol. 111, no. 4, pp. 2287–2310, 2020.

[18] M. Wang, T. Zhu, T. Zhang, J. Zhang, S. Yu, and W. Zhou, "Security and privacy in 6G networks: new areas and new challenges," *Digital Communications and Networks*, vol. 6, 2020.

[19] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582, Las Vegas, NV, USA, June 2016.

[20] N. Dalvi, P. Domingos, S. Sanghai et al., "Adversarial classification," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 99–108, Washington, DC, USA, July 2004.

[21] D. Lowd and C. Meek, "Adversarial learning," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 641–647, San Diego, CA, USA, August 2005.

[22] G. Lin, S. Wen, Q.-L. Han, J. Zhang, and X. Yang, "Software vulnerability detection using deep neural networks: a survey," *Proceedings of the IEEE*, vol. 108, 2020.

[23] M. Barreno, B. Nelson, R. Sears et al., "Can machine learning be secure," in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, pp. 16–25, Taipei, Taiwan, March 2006.

[24] N. Papernot, P. McDaniel, I. Goodfellow et al., "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 506–519, Abu Dhabi, UAE, April 2017.

[25] L. Huang, A. D. Joseph, B. Nelson et al., "Adversarial machine learning," in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pp. 43–58, Abu Dhabi, UAE, June 2011.

[26] I. M. Bapiyev, B. H. Aitchanov, I. A. Tereikovskyi et al., "Deep neural networks in cyber attack detection systems[J]," *International Journal of Civil Engineering and Technology (IJCIET)*, vol. 8, no. 11, pp. 1086–1092, 2017.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5593435, 9 pages
https://doi.org/10.1155/2021/5593435

Research Article

A Residual Learning-Based Network Intrusion Detection System

Jiarui Man[1,2] **and Guozi Sun**[1,2]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*Key Laboratory of Urban Land Resources Monitoring and Simulation, MNR, Shenzhen 518000, China*

Correspondence should be addressed to Guozi Sun; sun@njupt.edu.cn

Received 9 January 2021; Revised 13 February 2021; Accepted 5 March 2021; Published 16 March 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Jiarui Man and Guozi Sun. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Neural networks have been proved to perform well in network intrusion detection. In order to acquire better features of network traffic, more learning layers are necessarily required. However, according to the results of the previous research, adding layers to the neural networks might fail to improve the classification results. In fact, after the number of layers has reached a certain threshold, performance of the model tends to degrade. In this paper, we propose a network intrusion detection model based on residual learning. After transforming the UNSW-NB15 data set into images, deeper convolutional neural networks with residual blocks are built to learn more critical features. Instead of the cross-entropy loss function, the modified focal loss is calculated to address the class imbalance problem in the training set and identify minor attacks in the testing set. Batch normalization and global average pooling are used to avoid overfitting and enhance the model. Experimental results show that the proposed model can improve attack detection accuracy compared with existing models.

1. Introduction

With the continuing expanding network scale, network security confronts more sophisticated threats than ever before. Hence, network security issues are attracting increasing attention. Commonly used network security systems that discover suspicious attacks involve firewalls, intrusion detection systems (IDSs), and intrusion prevention systems (IPSs) [1]. Among them, the task of IDSs is to collect and identify abnormal behaviors in the network [2]. By analyzing captured data packets, IDSs can check legitimate network behaviors, detect the attacks, and report the attacks for further containment.

Conventional IDSs tend to get low detection rates and high false positive rates due to their reliance on patterns of known attacks. Researchers have applied artificial intelligence (AI) algorithms in the designing part of IDSs to provide better performances. The performance of an IDS is closely related to the selected classifier, while traditional machine learning algorithms tend to perform poorly in the scenarios when large amounts of network data packets are included. In recent years, deep learning has achieved outstanding results in multiple fields. The advantage of deep learning is that it can learn the hierarchical features from a large amount of data to improve model efficiency [3]. The application of deep learning can reduce costs of IDSs and strengthen the abilities to identify attacks.

Convolutional neural networks (CNNs) can extract deep and critical features from the given data. It is a general perspective that increasing the number of network layers can help learn better features; hence, the performance of model is improved. However, simply stacking more layers may fail these tasks. Furthermore, after the number of layers has reached a certain threshold, it may even lead to performance degradation. Residual learning is proposed to address the issue above. Residual is the error between the actual value and the estimated value, and residual learning is originally derived from the residual representation in image recognition [4]. Residual learning is realized by establishing a direct connection between the input and the output. CNNs based on residual learning have achieved outstanding results in image recognition [5]. They are easier to train and optimize than common CNNs. In network intrusion detection, it is also vital to build deeper networks to improve the detection capabilities of IDSs. Because residual learning allows CNNs to be deeper, this paper introduced the concept of residual learning into IDSs.

UNSW-NB15, an imbalanced network intrusion detection data set, is selected to evaluate the model. In real-time network intrusion detection, the class imbalance problem seriously affects the classification results [6]. Prediction models that predict only the dominant classes fail to identify the minor classes. Resampling techniques are common solutions to class imbalance problems. However, resampling techniques have their disadvantages. Oversampling might disrupt the original data, and it takes more time to train the model when using oversampling techniques. Undersampling might cause the loss of vital information, affecting the classification capabilities. Focal loss was originally proposed to balance the loss between samples. We apply the modified focal loss function in the proposed model to enhance the abilities to detect minor classes without disrupting the training data.

The major contributions of the paper can be summarized as follows:

(1) Propose a deep learning-based intrusion detection model with a higher accuracy compared with other existing models

(2) Introduce residual learning into the model to address the network degradation problem, allowing the model to learn deeper features

(3) Use a modified focal loss function to deal with the class imbalance problem in the training set

This paper is organized as follows:

(1) The first chapter gives a brief overview of network intrusion detection and the motivation of the proposed methodology

(2) The second chapter introduces the related work

(3) The third chapter provides the methodology and implementation process in detail

(4) The fourth chapter carries out the experiments and analyzes the testing results

(5) The final chapter concludes the paper

2. Related Work

Data preprocessing is a key step in network intrusion detection. It can extract key features that have great influences on the classification results, effectively reducing the size of data and improving the efficiency of given classifiers. Zhang et al. [7] proposed an effective network traffic classification method, which used principal component analysis (PCA) to remove the irrelevant features and applied Gaussian Naive Bayes as the classifier. Kasongo et al. [8] applied a filter-based feature reduction technique on the UNSW-NB15 data set using the XGBoost algorithm and then implemented several algorithms to classify the data. Results demonstrated that the feature selection method increased the test accuracy. Sun et al. [9] proposed an improved Naive Bayesian learning method which took the influence of different features into account. It achieved a higher accuracy than traditional machine learning algorithms. It can be seen that the performance of traditional classifiers is excessively dependent on the extracted features. However, traditional machine learning algorithms are shallow learning algorithms which require feature engineering. To build the fittest model, optimization of parameters is also needed. The size of the data set also affects the efficiency of the models. These difficulties slow down the training process of traditional machine learning algorithms and affect the overall network security.

In recent years, deep learning models have been gradually applied to intrusion detection to enhance the classification classifiers due to their high efficiency and easy implementation. Among deep learning models, CNNs have made great success in many fields [10–12], and researchers have applied CNNs in intrusion detection. Qian et al. [13] analyzed the network traffic with a CNN. In the training phase, rectified linear unit (ReLU) served as the activation function and adaptive moment estimation (Adam) algorithm was used to optimize the model. Lai et al. [14] also used a CNN as the intrusion detection model, achieving a higher accuracy rate than other deep learning models.

In the aspect of residual learning, the concept of Residual Network (ResNet) was proposed by He et al. [15] from Microsoft Research Institute to deal with the performance degradation problem as the number of layers grows. ResNets have outperformed common CNNs in image classification and object recognition [16–18]. Because of residual learning, the depth of ResNets is deeper than that of the traditional CNNs. In network intrusion detection, a deeper CNN can extract more critical features and get better classification results. Therefore, CNNs based on residual learning have been attempted in network intrusion detection. Wu et al. [19] proposed a deep neural network built upon residual blocks to discover malicious network behaviors, achieving a low false alarm rate. Chouhan et al. [20] proposed a multipath residual learning-based CNN architecture that was being evaluated on NSL-KDD data set, showing significant improvements over the previous research.

However, while residual learning can improve the overall performance of CNNs, in the practical aspect, it does not improve models' abilities to detect minor attacks due to lack of original training data. Classes in most modern network intrusion detection data sets are imbalanced. Therefore, most IDSs fail to provide better performances for attacks with fewer samples. Focal loss was proposed by He et al. [21] in 2017. Focal loss takes the different level of training difficulty of samples into consideration and focuses more on the difficult-to-train samples; therefore, it has been applied in many fields, such as object detection, imbalanced data classification, and so on [22–24]. To identify classes with fewer training samples more accurately, a modified focal loss function is used to replace cross-entropy loss function in the proposed model.

Choosing a suitable data set is vital for the building of IDSs. In recent years, most commonly used public data sets in network intrusion detection are KDD99 [25], NSL-KDD [26], and UNSW-NB15 [27]. In spite of being the most popular data sets in network intrusion detection, KDD99 and NSL-KDD are out-of-date due to old and redundant data. Evaluating IDSs using KDD99 and NSL-KDD does not reflect satisfactory results due to their shortcomings. According to the previous research and statistical analysis, compared with the other two data sets, UNSW-NB15 has more complex feature sets, contains more modern normal traffic scenarios, covers richer types of attack traffic, and contains fewer incomplete samples. Also, most new cyberattacks are variants of these known attacks in the UNSW-NB15 data set. Therefore, UNSW-NB15 can more accurately reflect the characteristics of modern network traffic data and is more suitable for evaluating IDSs. Therefore, we choose UNSW-NB15 data set as the evaluation set for the proposed model.

In summary, with the powerful capabilities of automatic feature extraction, deep learning has been applied to network intrusion detection. However, how to build deeper networks without triggering the performance degradation problem and address the class imbalance problem in the training set are two major challenges. In this paper, a residual learning-based CNN is constructed to learn deeper features of network traffic, and the modified focal loss function is introduced into the proposed model to detect minor attacks.

3. Methodology

The proposed methodology consists of three parts: data preprocessing, model constructing, and model evaluation. First, network flows are converted into images. Then, CNNs with residual learning are constructed. Finally, trained models are tested and evaluated. The main structure is shown in Figure 1.

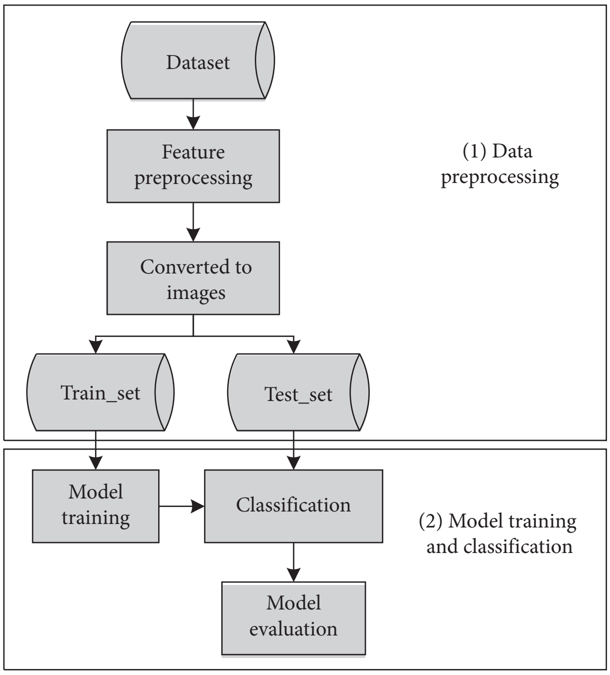


Figure 1: Model structure.

3.1. Data Set. As stated before, UNSW-NB15 is a network intrusion detection data set, which is processed and built through collecting different types of network connection data. This data set includes multiple types of contemporary attacks. Each flow of the data set contains 47 features, and the data set divides the network behaviours into nine categories of attacks plus the category of normal behaviours. These attacks can also be divided into 177 categories according to the environments that the specific attack depends on.

In this paper, part of the data set known as the UNSW-NB15 training set and UNSW-NB15 testing set are selected as the training data and testing data. They are data sets which are used for intrusion detection after redundant flows and features are processed. The distribution is shown in Table 1.

Table 1: Distribution of UNSW-NB15 training set and testing.

Type	Train_set	Test_set	Label
Analysis	2000	677	0
Backdoors	1746	583	1
DoS	12264	4089	2
Exploits	33393	11132	3
Fuzzers	18185	6062	4
Generic	40000	18871	5
Normal	56000	37000	6
Recon	10492	3496	7
Shellcode	1133	378	8
Worms	130	44	9

3.2. Data Preprocessing. Features in UNSW-NB15 contain numeric features and symbolic features; therefore, symbolic features should be digitized first. Then, processed features are normalized to obtain a standardized data set and converted into matrices.

One-hot encoding is used to map symbolic features into numerical features, and labels are mapped into digits using label encoding. The specific implementations are as follows:

(1) *One-Hot Encoding*. One-hot encoding mainly uses a state register of size X to encode a character, and each character will have an independent register bit

(2) *Label Encoding*. Labels in the UNSW-NB15 data set are divided into 10 categories. Coding rules are shown in Table 1

This paper uses Min-Max normalization. The main function of Min-Max normalization is to unify the feature values in the interval of [0,1]:

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \tag{1}$$

where x^* is the normalized eigenvalue, x is the original eigenvalue, $x_{\min}$ is the minimum eigenvalue, and $x_{\max}$ is the maximum eigenvalue. After numerical normalization, each flow of the new set contains 196 features, so the data are converted into 14 * 14 matrices; then, the matrices are changed into black and white images.

3.3. Network Construction. Figure 2 shows the overall structure and the parameters of the CNN model. The proposed model extracts the features of input data by the convolution layers and pooling layers. Feature maps are then input into a global average pooling layer. Finally, the model classifies the sample data with a softmax layer.

3.3.1. Convolution and Pooling. Convolution layers are the core parts of CNN models. Convolution layers in proposed model extracted spatial features of given data and produced a feature map as the output. ReLU is often used as the activation function:

$$f(x) = \max(0, x), \tag{2}$$

and the function of pooling layers is to reduce the size of feature maps.

3.3.2. Batch Normalization. In the training part of CNNs, with the change of the parameters of the previous layer, the input distribution of the next layer will change correspondingly, making it more difficult to train deeper neural networks. Batch normalization, in the training process of CNNs, makes the input of each layer maintain the same distribution and provides with the solution to the difficulty of network training, thus effectively improving the training speed of networks and avoiding overfitting. Input data are all divided into batches, for instance, parameter "batch_size" is set to 128; therefore, 128 pieces of data are input as a batch at a time. Batch normalization layers are to normalize each batch so that the distribution of data remains unchanged. Suppose we have a batch of inputs:

$$x = \{x_1, x_2, \ldots, x_n\}. \tag{3}$$

The output of batch normalization is computed by

$$y_i = \lambda * x_i' + \varphi, \tag{4}$$

where λ and φ are learned parameters and x_i' is calculated through

$$\begin{aligned}
\mu_\beta &= \frac{1}{m}\sum_{i=1}^{m} x_i;\\
\sigma_\beta^2 &= \frac{1}{m}\sum_{i=1}^{m}\left(x_i - \mu_\beta\right)^2;\\
x_i' &= \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \varepsilon}}.
\end{aligned} \tag{5}$$

In this paper, batch normalization layers are placed after convolution layers and before the activation functions.

Layer	Parameters
Convolution	3 * 3 * 64, stride 1
Maxpooling	3 * 3, stride 2
Residual block_1	Conv 1 * 1 * 64, stride 1 Conv 3 * 3 * 64, stride 1 Conv 1 * 1 * 256, stride 1
Residual block_2	Conv 1 * 1 * 128, stride 1 Conv 3 * 3 * 128, stride 1 Conv 1 * 1 * 512, stride 1
Residual block_3	Conv 1 * 1 * 256, stride 1 Conv 3 * 3 * 256, stride 1 Conv 1 * 1 * 1024, stride 1

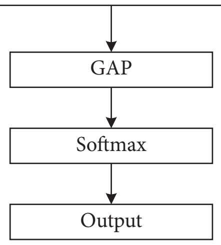


Figure 2: CNN model.

3.3.3. Residual Learning. Compared with common CNNs such as the LeNet-5 [28] and the AlexNet [29], ResNets introduced residual learning into the constructing of CNNs. The depth of a CNN has a great influence on the final classification results, so deeper networks are often constructed. However, as the network depth increases, the phenomenon of gradient explosion might occur, and the performance of the network will degrade. According to the previous experimental results, simply adding convolution layers and pooling layers to the network does not improve the accuracy of the network but leads to the deterioration of network performance. In this paper, residual learning is used to address the issue above. Residual refers to the residual difference between the local input and output:

$$f(x) = g(x) - x. \tag{6}$$

In contrast to identity mapping, the learning goal of residual learning is 0, that is, to reduce the difference between the input and the output, allowing the original input to be directly connected to one certain network layer, so that the network can learn the residual. Residual learning is realized by a fast shortcut connection between the input and output of a block. It not only avoids adding additional parameters and computations to the network, but also effectively trains the parameters in the network and guarantees the performance while the network can learn deeper features. Two blocks used in this paper are shown in Figure 3. In the construction of plain models, the normal block exhibited in (a) is used, while the residual block shown in (b) is used to construct residual networks.

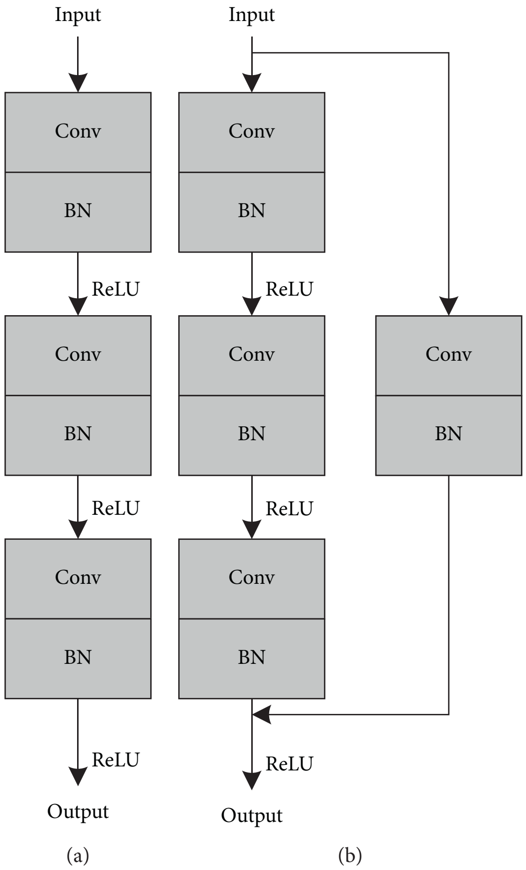


FIGURE 3: Normal block and residual block. (a) Normal block. (b) Residual block.

3.3.4. Global Average Pooling. At the end of CNN models, flatten layers are often adopted to flatten the data processed by the previous layers into a one-dimensional vector. The output size is gradually reduced through full connection layers, and the final output is obtained through an activation function. Since every node in flatten layers and full connection layers is connected to each other, too many parameters may lead to overfitting. A global average pooling layer is an average pooling layer without filter size. It averages the entire feature map. Using a global average pooling layer can reduce the count of calculating parameters and accordingly reduce the possibility of overfitting. In this paper, a global average pooling layer is used to replace the flatten layer. The principle of global average pooling is shown in Figure 4.

3.3.5. Softmax and Loss Function. Finally, the probability distribution of each label is calculated through the softmax layer:

$$S_j = \frac{e^{a_j}}{\sum_{k=1}^{N} e^{a_k}}, \tag{7}$$

where N denotes the total count of classes.a_j denotes the j_{th} input of softmax layer. Cross-entropy loss function is defined as

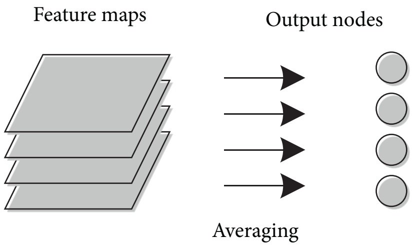


FIGURE 4: Global average pooling.

$$L = -\sum_{k=1}^{N} y_k \log S_k, \tag{8}$$

where y_k denotes the probability that tested sample belongs to class k.

With the obvious class imbalance problem in the training set, preventing loss function from optimising one category while suppressing other categories is important. To increase the classification accuracy for minor classes, we need to make the model pay more attention to them during training. Resampling is one of the most common methods to deal with imbalanced data. Among resampling methods, undersampling may cause the loss of vital information while oversampling may add new information to disrupt the data and greatly increase training time. Compared with cross-entropy loss function, focal loss aims to solve the class imbalance problem so that if the number of samples that are easy to train is large, contribution of certain samples to the total loss is small. In other words, focal loss function focuses on minor samples. In our multilabel classification, focal loss is defined as

$$FL_{\text{loss}} = -a_t (1 - p_t)^{\gamma} \log(p_t), \tag{9}$$

where $(1 - p_t)^{\gamma}$ is a modulating factor that reduces loss contribution from easy samples.p_t was calculated through

$$p_t = \begin{cases} p, & y = 1 \\ 1 - p, & \text{otherwise}, \end{cases} \tag{10}$$

where $p \in [0, 1]$ represents the category prediction probability and y is the label value. As $p_t \longrightarrow 1$, $(1 - p_t)^{\gamma}$ goes to 0 and the weights of samples that are easy to train to the loss are reduced. And a_t is a weighting factor that can be used to scale the minor classes separately. In this paper, we introduce the multilabel focal loss where γ was set to 2 and a_t was calculated through

$$a_t = 1 - \frac{\text{num}_t}{\text{total\_cnt}}, \tag{11}$$

where num_t denotes the number of samples belonging to class t and total_cnt denotes the total number of samples in the training data.

4. Experiments

4.1. Experimental Environments. Experimental environments of this paper are shown in Table 2.

Table 2: Experimental environment.

Environment	Value
OS	Windows
CPU	i7-7700 HQ
Memory	DDR4 8 GB
Language	Python
SDE	Keras
Tool	Anaconda

4.2. Evaluation Metrics. Accuracy, precision, recall, and F1-measure are adopted as evaluation metrics.

(1) Accuracy (Acc): the ratio of the number of correctly classified samples to the total number of samples tested.

(2) Precision (P): the ratio of correctly classified positive samples to the total number of positive samples.

(3) Recall (R): the ratio of accurately identified positive samples to the total number of positive samples in the testing set.

(4) F1-measure (F1): the weighted average of precision and recall.

$$\begin{aligned} \text{accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}; \\ \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}; \\ \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}; \\ F1 &= \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \end{aligned} \tag{12}$$

True Positive (TP) denotes the number of positive samples correctly classified as positive; True Negative (TN) denotes the number of negative samples correctly classified as negative; False Positive (FP) denotes the number of negative samples misclassified as positive; and False Negative (FN) denotes the number of positive samples misclassified as negative. The confusion matrix is shown in Table 3.

4.3. Experimental Performance Evaluation. In the training phase, 6 CNN models are constructed, including 3 plain models and 3 residual models. Model P_n (R_n) consists of n normal blocks (residual blocks) shown in Section 3.3.3. Each model is trained by the processed training set for 100 epochs. The learning rate is set to 0.01. And after calculating the loss, an optimizer is needed to update the network weights. Adam is selected as the optimizer. Performances of 6 models are evaluated by calculating the model accuracy and the weighted average of precision, recall, and F1-measure. We choose weighted average to evaluate the overall performance, because compared with other average methods like the micro average and the macro average, the weighted average method takes the number of samples belonging to each class into consideration, so its results are more convincing to reflect the performance of the model. The weighted average is defined as

Table 3: Confusion matrix.

Actual class	Predicted class	
	Positive	Negative
Positive	TP	FN
Negative	FP	TN

$$W_M = \sum_{i=1}^{k} n_k M_k, \tag{13}$$

where k denotes the total amount of classes, n_k denotes the number of testing samples in class k, and M_k is the testing result of metric M on class k.

We record the training loss of the above 6 models every 20 epochs to examine the effects of residual learning on CNNs. It can be seen from Figure 5, by utilizing residual learning in the blocks, training loss is greatly reduced. By adding residual blocks in the CNN, we achieve lower training loss, indicating that residual learning can address the network degradation problem. Also, as the figure demonstrates, the training loss at the very beginning is quite large, but as the training process progresses, the loss value continues to decrease. When the training epoch reaches 20, the training loss tends to decrease at a slower rate.

According to the comparison results from Table 4, the overall performance of CNNs has been significantly improved with residual blocks added into the plain models, indicating that we can build deeper CNNs with residual learning. Results can also demonstrate that with the increasing number of network layers, residual networks can achieve better performances than shallow residual networks on the whole. The model with 3 residual blocks (R3) achieves the highest overall classification accuracy of 88.695%. R3 (RLF-CNN) will be further compared with the state-of-the-art classification algorithms.

In order to evaluate the abilities of proposed method to detect attacks like Shellcode and Worm in network intrusion detection, we conduct several experiments and compared the recall value of each class with Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM), LeNet-5, AlexNet, and CNN with simple cross-entropy loss function (RLC-CNN). Support Vector Machine (SVM) and Random Forest (RF) are commonly used machine learning algorithms in network intrusion detection [30, 31]. We select SVM and RF to compare their classification results with those of deep learning algorithms. The number of training epochs of deep learning models was also set to 100. Table 5 and Figure 6 demonstrate the recall values of all models on each class. Figure 6 also shows the overall accuracy of each model. It can be seen from Table 5 that the performance of deep neural networks is significantly better than the classic machine learning algorithms. Classic machine learning algorithms need manually designed features of network traffic before the training phase, while deep learning algorithms automatically extract features.

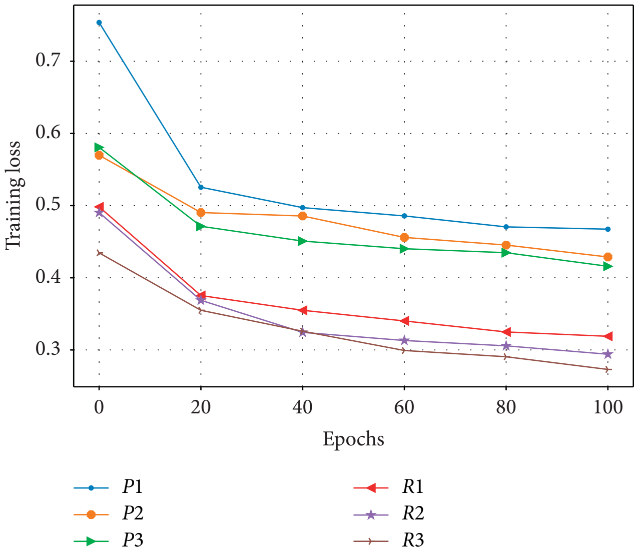


Figure 5: Training loss.

Table 4: Classification results of 6 models.

Model	Testing metrics			
	Acc	P_W	R_W	$F1_W$
P1	0.849	0.861	0.843	0.825
P2	0.869	0.852	0.873	0.859
P3	0.864	0.853	0.868	0.844
R1	0.872	0.864	0.875	0.851
R2	0.881	0.869	0.884	0.865
R3	0.887	0.875	0.893	0.879

Table 5: Comparison of results of recall with other classifiers.

Model	Testing metric R									
	Analysis	Backdoors	DoS	Exploits	Fuzzers	Generic	Normal	Recon	Shellcode	Worms
RF	0	0	0.012	0.839	0.741	0.630	0.999	0.754	0.111	0
SVM	0.062	0.033	0.183	0.669	0.730	0.763	0.997	0.645	0.206	0.068
MLP	0.007	0.039	0.076	0.825	0.679	0.891	0.999	0.661	0.508	0.227
LSTM	0	0	0.008	0.853	0.793	0.747	0.998	0.683	0	0.045
LeNet-5	0	0	0	0.852	0.689	0.901	0.996	0.787	0.553	0
AlexNet	0	0.002	0.057	0.833	0.725	0.915	0.999	0.711	0.352	0.114
RLC-CNN	0	0.026	0.077	0.875	0.711	0.926	0.998	0.805	0.370	0
RLF-CNN	**0.435**	**0.482**	**0.311**	**0.860**	**0.732**	**0.918**	**0.993**	**0.797**	**0.712**	**0.818**

Among deep learning algorithms, we are able to build deeper networks to learn more critical features of network traffic due to the residual blocks. Table 5 and Figure 6 demonstrate that our model achieves better results than other deep learning algorithms. With residual learning, CNNs can provide better performances in network intrusion detection.

In terms of minor classes, all the other models perform poorly due to the class imbalance problems in the training set. Our model utilizes focal loss to address the issue above. Although in some dominant classes, RLF-CNN's performance slightly weakens due to their reduced weights in the loss function, RLF-CNN outperforms other classifiers in the classification of minor classes with higher recall values, indicating that focal loss is more suitable in classifying imbalanced data sets and enhancing the detecting capabilities.

To prove our model's ability to detect normal flows and attacks, we compare it with other algorithms using metrics including True Positive and True Negative. The testing results are shown in Table 6.

Among these models, RLC-CCNN is the improved version of RLC-CNN possessing the same class weights as the ones used in the focal loss of RLF-CNN. SMOTE-RF [32] is an algorithm of Random Forest combined with SMOTE. Pelican [19] and S-ResNet [1] are improved

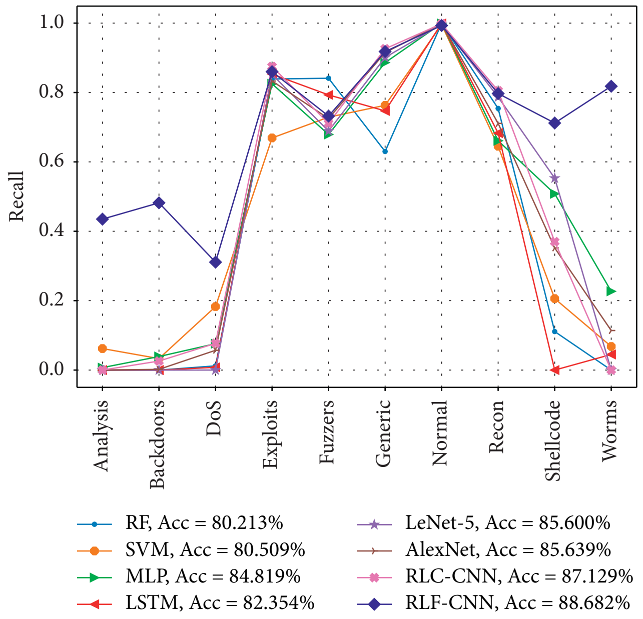


Figure 6: Recall and accuracy of classifiers.

Table 6: Comparison with other detection methods on UNSW-NB15 testing set.

Model	Testing metrics			
	TN	TP	Acc	R_W
RLF-CNN	36758	36278	0.887	0.893
RLC-CCNN	36996	35429	0.879	0.881
SMOTE-RF [32]	36952	32286	0.841	0.826
Pelican [19]	36850	34928	0.872	0.859
S-ResNet [1]	36928	31427	0.830	0.842

residual networks which have faster convergence velocity and better testing results compared to other deep learning algorithms. As shown in Table 6, all the models above can identify over 99.3% of all 37000 normal samples correctly. But compared with other contemporary algorithms for network intrusion detection, RLF-CNN can identify more attacks correctly, given that most of the attacks in the data set are minor samples, showing higher attack detection rates.

Compared with SMOTE-RF, our model detects more attacks while it avoids generating new data. SMOTE-RF generates over 300000 training samples, consuming a lot of time and memory. Also, tradition machine learning algorithms lack the abilities to acquire data features automatically; therefore, with the absence of feature engineering techniques, SMOTE-RF is inferior to others in detecting attacks. Compared with other residual networks, our model got better results in the detection of attacks. It can also be seen that our model outperforms RLC-CCNN. Focal loss enables the model to focus on samples that are harder to learn, and testing results indicate that the focal loss can learn complex samples more efficiently and is superior to class weights in the training phase.

5. Conclusions

In this paper, a network intrusion detection method based on residual learning and focal loss has been proposed. Experimental results show that models with residual learning are easier to train, achieving lower loss values on the training data and higher accuracy rates on the testing data. Compared with other deep learning algorithms, RLF-CNN has achieved better performance in terms of several metrics due to residual learning. And our model uses a modified focal loss function to deal with the class imbalance problem existing in the training data. Also, the proposed model shows better results than a CNN with the same class weights. Despite outstanding results, this study has its potential limitations. Although our model has outperformed other deep learning algorithms in the detection of minor attacks with the focal loss, its performance to detect some dominant classes has weakened due to reduced weights. Therefore, how to improve the model's performance on minor classes without affecting its abilities to detect dominant classes is an important issue that needs to be addressed in the future. Also, UNSW-NB15 data set only contains a few types of attacks; due to the low tolerance for errors in IDSs, we will combine other data sets to cover various types of attacks in the future. Last but not least, due to limited computing resources, deeper neural networks with more residual blocks and normal blocks cannot be tested. So, with more powerful resources in the future, we will continue to perform more experiments and maybe get better results when it comes to detecting network attacks.

Data Availability

The processed UNSW-NB15 data sets 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 publishing of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61906099) and the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (no. KF-2019-04-065).

References

[1] Y. Xiao and X. Xiao, "An intrusion detection system based on a simplified residual network," *Information*, vol. 10, no. 11, p. 356, 2019.

[2] F. Safara, A. Souri, and M. Serrizadeh, "Improved intrusion detection method for communication networks using association rule mining and artificial neural networks," *IET Communications*, vol. 14, no. 7, pp. 1192–1197, 2020.

[3] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, 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, 2012.

[5] Z. Chen, Y. Zhou, and Z. Huang, "Auto-creation of effective neural network architecture by evolutionary algorithm and ResNet for image classification," in *Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 3895–3900, IEEE, Bari, Italy, October 2019.

[6] 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. 2685–2688, IEEE, Chennai, India, March 2016.

[7] B. Zhang, Z. Liu, Y. Jia et al., "Network intrusion detection method based on PCA and Bayes algorithm," *Security and Communication Networks*, vol. 2018, Article ID 1914980, 11 pages, 2018.

[8] M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *Journal of Big Data*, vol. 7, no. 1, p. 105, 2020.

[9] C. Sun, J. Xing, Q. Yang, and D. Han, "Intrusion detection methods based on improved Naive Bayesian," *Microcomputer and its Applications*, vol. 2017, no. 1, pp. 8–10, 2017.

[10] J. Wang, X. Chen, L. Cao et al., "Individual rubber tree segmentation based on ground-based LiDAR data and faster R-CNN of deep learning," *Forests*, vol. 10, no. 9, p. 793, 2019.

[11] J. Pons and X. Serra, "Randomly weighted CNNs for (music) audio classification," in *Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 336–340, IEEE, Brighton, UK, May 2019.

[12] Y. Peng, M. Liao, H. Deng et al., "CNN-SVM: a classification method for fruit fly image with the complex background," *IET Cyber-Physical Systems: Theory & Applications*, vol. 5, no. 2, pp. 181–185, 2020.

[13] T. Qian, Y. Wang, M. Zhang et al., "Intrusion detection method based on deep neural network," *Journal of Huazhong University of Science & Technology*, vol. 46, no. 1, pp. 6–10, 2018.

[14] Y. Lai, J. Zhang, Z. Liu, and M. Alazab, "Industrial anomaly detection and attack classification method based on convolutional neural network," *Security and Communication Networks*, vol. 2019, Article ID 8124254, 11 pages, 2019.

[15] K. He, X. Zhang, S. Ren et al., "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, IEEE, Las Vegas, NV, USA, June 2016.

[16] M. Feng, H. Lu, and Y. Yu, "Residual learning for salient object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 4696–4708, 2020.

[17] P. McAllister, H. Zheng, R. Bond, and A. Moorhead, "Combining deep residual neural network features with supervised machine learning algorithms to classify diverse food image datasets," *Computers in Biology and Medicine*, vol. 95, pp. 217–233, 2018.

[18] T. Li, H. Song, K. Zhang, and Q. Liu, "Recurrent reverse attention guided residual learning for saliency object detection," *Neurocomputing*, vol. 389, pp. 170–178, 2020.

[19] P. Wu, H. Guo, and N. Moustafa, "Pelican: a deep residual network for network intrusion detection," in *Proceedings of the 2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 55–62, IEEE, Valencia, Spain, July 2020.

[20] N. Chouhan, A. Khan, and H. Khan, "Network anomaly detection using channel boosted and residual learning based deep convolutional neural network," *Applied Soft Computing Journal*, vol. 83, 2019.

[21] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2020.

[22] S.-H. Choi and S. H. Jung, "Stable Acquisition of fine-grained segments using batch normalization and focal loss with L1 regularization in U-Net structure," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 20, no. 1, pp. 59–68, 2020.

[23] C. Wang, C. Deng, and S. Wang, "Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost," *Pattern Recognition Letters*, vol. 136, pp. 190–197, 2020.

[24] Y. Zhao, F. Lin, S. Liu, Z. Hu, H. Li, and Y. Bai, "Constrained-focal-loss based deep learning for segmentation of spores," *IEEE Access*, vol. 7, pp. 165029–165038, 2019.

[25] V. Abhishek and R. Virender, "Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning," *Procedia Computer Science*, vol. 125, pp. 709–716, 2018.

[26] C. Sarika and K. Nishtha, ""Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT," *Procedia Computer Science*, vol. 167, pp. 1561–1573, 2020.

[27] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1–3, pp. 18–31, 2016.

[28] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[29] 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.

[30] L. Liu, P. Wang, J. Lin, and L. Liu, "Intrusion detection of imbalanced network traffic based on machine learning and deep learning," *IEEE Access*, vol. 9, pp. 7550–7563, 2021.

[31] Y. Chang, W. Li, and Z. Yang, "Network intrusion detection based on random forest and support vector machine," in *Proceedings of the 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, pp. 635–638, IEEE, Guangzhou, China, July 2017.

[32] X. Tan, S. Su, Z. Huang et al., "Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm," *Sensors*, vol. 19, no. 1, p. 203, 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6670847, 16 pages
https://doi.org/10.1155/2021/6670847

Research Article

An Efficient Anonymous Communication Scheme to Protect the Privacy of the Source Node Location in the Internet of Things

Fengyin Li,[1] **Pei Ren**,[1] **Guoyu Yang**,[1] **Yuhong Sun**,[1] **Yilei Wang**,[1] **Yanli Wang**,[1] **Siyuan Li**,[1] and **Huiyu Zhou**[2]

[1] *School of Computer Science, Qufu Normal University, Rizhao 276826, China*
[2] *School of Informatics, University of Leicester, Leicester LE1 7RH, UK*

Correspondence should be addressed to Yilei Wang; wang_yilei2019@qfnu.edu.cn

Received 18 December 2020; Revised 22 January 2021; Accepted 15 February 2021; Published 24 February 2021

Academic Editor: Wenjuan Li

Copyright © 2021 Fengyin Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Advances in machine learning (ML) in recent years have enabled a dizzying array of applications such as data analytics, autonomous systems, and security diagnostics. As an important part of the Internet of Things (IoT), wireless sensor networks (WSNs) have been widely used in military, transportation, medical, and household fields. However, in the applications of wireless sensor networks, the adversary can infer the location of a source node and an event by backtracking attacks and traffic analysis. The location privacy leakage of a source node has become one of the most urgent problems to be solved in wireless sensor networks. To solve the problem of source location privacy leakage, in this paper, we first propose a proxy source node selection mechanism by constructing the candidate region. Secondly, based on the residual energy of the node, we propose a shortest routing algorithm to achieve better forwarding efficiency. Finally, by combining the proposed proxy source node selection mechanism with the proposed shortest routing algorithm based on the residual energy, we further propose a new, anonymous communication scheme. Meanwhile, the performance analysis indicates that the anonymous communication scheme can effectively protect the location privacy of the source nodes and reduce the network overhead.

1. Introduction

The coming of age of the science of machine learning (ML) coupled with advances in computational and storage capacities have transformed the technology landscape. For example, within the security domain, detection and monitoring systems now consume massive amounts of data and extract actionable information. ML is now pervasive—new systems and models are being deployed in every domain imaginable [1]. Internet of Things is based on the Internet, using RFID, wireless data communication, and other technologies to construct a network covering possible nodes in the world. In the IoT, objects can "communicate" with each other without human intervention. IoT devices are committed to maximizing their utility within a limited capacity and maintaining the security of the IoT system [2]. Wireless sensor networks are an important part of the Internet of Things, and it is a distributed sensor network. Sensor nodes are stationary or mobile, and they constitute a wireless sensor network in a self-organizing and multihop manner. As a link between the physical world and the virtual world, the WSNs have become one of the most promising technologies. They have the ability to monitor object exits in the network, and realize data collection, processing, and transmission. At present, the WSNs have been widely used in military, transportation, medical, household, and industrial fields. However, due to the large-scale deployment of wireless sensor networks, security and privacy risks become critical. Currently, in the context of the Internet of Things, the WSNs still face security problems. For instance, adversaries can infer the location information of the source node through backtracking attacks, and then obtain the location information of an event, which causes the leakage of sensitive messages.

In the IoT, many existing security schemes can protect message content and contextual information using the traditional cryptography theory, but they cannot solve the problem of the source node location privacy protection.

Adversaries can obtain sensitive messages through traffic analysis [3], hop-by-hop tracking, backtracking attacks, and other methods. Moreover, many schemes fail to take into account the finiteness of the node resources, which result in a large amount of resource consumptions, unsuitable for the WSNs with limited energy. In addition, many schemes cannot resist traffic analysis attacks. A large amount of traffic will be generated near the source nodes due to data packets being transmitted between nodes. Therefore, when an adversary analyzes the traffic in the network to find the hot spots in the network, it will obtain the correct location of the source node and attack it.

In the random routing algorithm, after purely random h hops, the probability that the proxy source node is no more than $h/5$ hops away from the real source node is as follows [4]:

$$P = 1 - e^{-(h/25)}. \tag{1}$$

When h is large enough, the value of P approaches 1. In other words, purely random routing does not guarantee that the selected proxy source node is far enough away from the real source node. However, if the selected proxy source node is very close to the real source node, the position of the real source node cannot be effectively hidden. Therefore, the proxy source node that we select is required to be far enough away from the real source node.

In this paper, we propose an efficient anonymous communication scheme to protect the privacy of the source node location, which not only protects the source location privacy in wireless sensor networks but also guarantees the forwarding efficiency of anonymous messages through the shortest path routing algorithm, and reduces the network overhead.

Specifically, the main contributions of this paper are as follows:

(1) We propose the mechanism for selecting proxy source nodes based on the candidate region. This proposal selects the nodes that meet the upper and the lower limits of the hop count to construct the candidate region around the source node. It also selects the proxy source node to replace the real source node in the candidate region to forward the messages in order to realize the location privacy protection of the real source node.

(2) We propose the shortest path routing algorithm based on the residual energy. When forwarding a message from the proxy source node to the sink, each node selects the next hop according to the residual energy of its neighbor nodes and the minimum number of hops from the sink to itself. We propose the shortest path routing algorithm based on the residual energy, which improves the efficiency of anonymous message forwarding.

The rest of the paper is organized as follows. In Section 2, we introduce related work. In Section 3, we first design the system model of the anonymous communication system, and then we propose an anonymous communication scheme based on the proxy source node and the shortest path routing. In Section 4, the results and discussion of the proposed scheme is given. Finally, we conclude this paper in Section 5.

2. Related Works

In recent years, privacy issues have been a hot issue in machine learning. To avoid privacy issues caused by massive data collection, Mohassel et al. proposed new and efficient protocols for privacy preserving machine learning for linear regression [5], logistic regression, and neural network training using the stochastic gradient descent method. In order to solve the risk of large-scale collection of sensitive data, Bonawitz et al. designed a novel, communication-efficient, failure-robust protocol for the secure aggregation of high-dimensional data [6]. The protocol has good security in an honest but curious and active adversary environment. At the same time, even if a randomly selected subset of users exits at any time, the security is maintained.

Blockchain has been widely discussed and used in the IoT [7, 8], in order to be able to balance between fairness and incentive compatibility. Wang et al. tailored a new bonus reward function by adding random salts to the geometric reward function [9]. Li et al. highlighted the combination of game theory and blockchain [10], including rational smart contracts, game theoretic attacks, and rational mining strategies. In the IoT, privacy issues are also a hot issue of research. The service evaluation model is an important part of the service-oriented Internet of Things (IoT) architecture, but it is vulnerable to various attacks. Li et al. put forth a new service evaluation model named Tesia allowing specific users to submit the comments as a group in the IoT networks [11] to solve the problem. To enhance the privacy of the source location in wireless sensor networks, Zhao et al. conducted a comprehensive investigation on the theory and practice of the SMPC protocol [12], explaining the security requirements and the basic construction technology of the SMPC. It also introduces the research progress of the general SMPC protocol construction technology and its application in the IoT. Wang et al. proposed a trace-cost-based source location privacy protection scheme in wireless sensor networks for a smart city (TCSLP) [13], by constructing a phantom area, and combining shortest path routing and random routing to send packets, whereby the security time of the smart city in the wireless city is extended and the SLP is enhanced. Zhu proposed a method of regional division based on node location information [14], and by using this method, he selects the hop distance between the location nodes. The distance accuracy of the data nodes in the vicinity selected during information transmission is improved, and the location privacy of the source node is better protected. Han et al. proposed a dynamic ring-based routing (DRBR) scheme [15], which solved the balance issues between security and energy consumption and provided efficient source location privacy. Muruganathan et al. proposed a centralized energy-efficient routing protocol for the WSNs (BCDCP) [16], which can evenly distribute energy consumption among all the sensor nodes to improve network life and save energy on average. Mutalemwa and Shin proposed a routing scheme with stronger source location privacy than the traditional routing scheme [17], providing a highly random routing path between the source and the sink nodes. Randomly send data packets to the sink node through tactically positioned proxy nodes, and implement the stronger source location privacy. In order to protect the privacy of the event and observe the privacy of the source node reporting the event, Chakraborty and Verma proposed a differential privacy framework [18]. By reporting the accumulation of the real and the virtual traffic of the same event, they distinguished the real and virtual events and provided differential privacy protection for nodes in the network.

Wang et al. proposed a data domain partitioning model [19], which is more accurate to choose the grid size. They proposed a uniform grid release method based on this model, and further improved the query accuracy. To solve the problem of privacy leakage caused by data analysis and mining, Spachos and Toumpakaris proposed a source-location privacy scheme that employs randomly selected intermediate nodes based on inclination angles [20], and analyzed the introduced angle-based dynamic routing scheme. However, as this scheme is for the data transmission of the included angle region, it could not adapt this angle for selecting an optimal routing. Furthermore, Liu and Xu proposed a new scheme to dynamically change the included angle in the ADRS—dynamic routing scheme (VADRS) based on the included Angle [21]. The scheme further improves the security performance of the ADRS by selecting the optimal Angle for data transmission at each hop. Aiming at the low security cycle of the existing source location privacy protection algorithm, Bai et al. proposed a source location privacy protection algorithm based on the expected phantom source node [22]. An ellipse is established through the coordinates of the source and the sink nodes, and a node is randomly selected on the ellipse as the expected phantom source node. The source location privacy protection is realized based on the phantom source node. Li et al. proposed a new routing strategy [23]. The routing strategy is divided into three stages to route data packets to the base station: directional random route, H-hop route, and the shortest path route in the ring area. The source location privacy protection is realized when information is sent to the base station in the WSNs.

Lin proposed schemes such as the ant colony algorithm to protect the location information of the source nodes and the multisource and the multipath protection of the source node location, etc. [24], to achieve the protection of node location privacy. To avoid the leakage of user personal information from the IoT devices during data processing and transmission, Li et al. proposed a certificateless encryption scheme to implement a novel anonymous communication protocol [25]. In the protocol, an anonymous communication link establishment method and an anonymous communication packet encapsulation format are proposed. It improves the privacy, security, and efficiency of CPSS anonymous communication. Sharma and Ghosh proposed new technology to prevent active and passive attacks in the mobile base station environment [26]. By deploying mobile sinks in the network, data were collected from sensor nodes and sent to the fixed base station, so as to guarantee the privacy of data in the mobile sink. Tan et al. proposed two effective source node location privacy protection policies [27]: the enhanced directional random routing protection mechanism (EDROW) and the multilayer ring proxy filtering mode routing protection mechanism (MRPFS). Aiming at the hop-by-hop reverse attackers with local traffic analysis behavior, Zhao et al. proposed the source location privacy protection routing protocol RAPFPR [28] based on the random angle and the probability forwarding. This protocol produces phantom nodes and enables them to be evenly distributed around the real source nodes and adopts the probabilistic forwarding routing mechanism, thus greatly reducing the generation of the overlapping paths. Sheu and Jiang proposed an anonymous path routing protocol (APR) for the wireless sensor networks [29]. This protocol encrypts data based on pair-wise keys, realizes the anonymous message transmission between adjacent nodes and the anonymous information transmission between the source node and the target node in the multi-hop communication path, and protects the data communication in the WSNs. Li and Ren proposed a source-location privacy scheme [30]. In this scheme, an anonymous path is constructed by randomly selecting intermediate nodes far away from the source node to realize the transmission of anonymous messages to the sink node. This solution provides satisfactory privacy of the local source location.

In order to better improve the privacy of source locations in the Internet of Things, we propose an anonymous communication scheme based on the proxy source node and the shortest path routing in this paper. This scheme can prevent the adversary from obtaining the location of the source node and event by means of backtracking attacks and traffic analysis. At the same time, the shortest path routing algorithm in this paper takes into account the residual energy of each node, ensuring the rationality of the energy overhead of the whole network.

3. An Anonymous Communication Scheme Based on the Proxy Source Node and the Shortest Path Routing

In order to realize the privacy protection of source locations in the IoT, we propose an anonymous communication scheme to protect the privacy of the source node location. In this anonymous communication scheme, the privacy protection of the source node location is achieved by setting the candidate region to select the proxy source node; the shortest routing algorithm based on the residual energy is used to achieve efficient anonymous message forwarding.

3.1. System Model

3.1.1. Network Model. First, we make the following assumptions about the network model:

① The wireless sensor network is composed of sensor nodes that are uniformly and randomly deployed, which cannot be moved at will after the nodes have been deployed. Any two nodes can communicate through multihop [31].

② The appearance of the object is randomly distributed throughout the network, so the probability of each sensor detecting the object information is equal. The node that detects the object, i.e., the source node, periodically generates data packets and sends them to the base station. There is only one base station in the whole network, the base station is safe, and it cannot be destroyed by adversaries.

③ The adversary cannot attack the object in the area that is one hop away from the base station because this area has powerful surveillance capabilities.

The symbols used in this paper are shown in Table 1.

3.1.2. Adversary Model. The adversary is assumed to be an external, passive, and global attacker [31]:

① *External.* An external adversary is an attacker who will not compromise or control any sensor nodes.

② *Passive.* Passive means that we assume that the adversary will not conduct any active attacks, such as traffic injection, channel interference, or denial of service attack. The adversary cannot decrypt the data packet and tamper with the contents of the data packet, nor destroy the sensor node.

③ *Global.* A global adversary is the one who we assume that an adversary can collect and analyze communications throughout the network.

3.1.3. Energy Consumption Model. In the IoT, no matter what routing strategy is used for data transmission, each node will consume energy to send and receive data. Therefore, here, we only consider the energy consumption generated when sending and receiving a certain number of bits of information [31, 32]. If the sender wants to send n-bit data to the receiver, and the distance between the two parties is l, then for the sender, the energy consumed to send n-bit data is defined as

$$E_{\text{sender}}(n, l) = \begin{cases} nE_{\text{con}} + n\varepsilon_{fs}l^2, & l < l_0, \\ nE_{\text{con}} + n\varepsilon_{\text{amp}}l^4, & l \geq l_0. \end{cases} \tag{2}$$

For the receiver, the energy consumed to receive n-bit data is defined as

$$E_{\text{receiver}}(n) = nE_{\text{con}}. \tag{3}$$

Among them, E_{con} represents the energy consumption in the sender or the receiver circuit, and the value of E_{con} is related to the distance between the sender and the receiver, i.e., l. We consider two models: for the free space and the multi-path fading channel models, their power losses are l^2 and l^4, respectively. ε_{fs} and ε_{amp} are the energies required by the power amplification in these two models, respectively.

3.1.4. DH Key Exchange Algorithm. When two parties communicate, the storage and disclosure of the user keys is a very important issue. We should ensure the identity privacy of both parties and the forward-backward security of the keys [33, 34]. Diffie–Hellman Key Exchange (D-H) is an algorithm jointly invented by Diffie and Hellman. Both parties in communication are able to generate shared cryptographic numbers only by exchanging publicly available information, and this cryptographic number is used as a key. This key can be used as a symmetric key to encrypt the communication content in subsequent communications.

Specifically, we assume that both Alice and Bob need a symmetric cryptographic key, but the communication line between the two parties has been eavesdropped on by an eavesdropper. At this time, Alice and Bob can generate the shared key by taking the *DH* key exchange in the following way:

① Take the prime number p and the integer a, a is a primitive root of p, a and p are disclosed

② Alice chooses a random number $X_A < p$, and calculates $Y_A = a^{X_A} \bmod p$

③ Bob chooses a random number $X_B < p$, and calculates $Y_B = a^{X_B} \bmod p$

④ Each party keeps X secret and Y public to the other party

⑤ The way Alice calculates the key is $K = Y_B^{X_B} \bmod p$

⑥ The way Bob calculates the key is $K = Y_A^{X_B} \bmod p$

In this way, Alice and Bob have the equal shared key.

3.2. Proxy Source Node Selection Mechanism Based on Candidate Region. The main idea of anonymous communication is to hide the identity or the communication relationship of the two parties through a certain method, so that the adversary cannot directly know or infer the communication relationship between the two parties or the party of the communication.

Anonymous communication in the WSN includes sender anonymity, receiver anonymity, and communication relationship anonymity. In this paper, we mainly focus on the sender's the anonymity and communication relationship anonymity. In order to hide the location information of the real source node, realize the privacy of the source location, and then realize the anonymity in WSN—the anonymity of the communication relationship, there is no identity information involved in the process of message transmission. Each node only knows who its previous hop and next hop are, and does not know the source and the destination of the information. At the same time, we must ensure that the selected proxy source node is far away from the real source node, so as to better protect the location privacy of the real source node.

We use the limited flooding from the real source node to establish an anonymous proxy path, and then establish a candidate region. Before each message is forwarded, a node will be selected from the candidate region as the proxy source node to send the message instead of the real source node. The real source node selects neighbor nodes that meet the energy requirements from the neighbor node list, and sends the detection data packets to the neighbor nodes.

Table 1: Definition of notations in this paper.

Notation	Definition
Rs	The real source node
Ps	The proxy source node
$E(\cdot)$	The secure encryption algorithm
$D(\cdot)$	The secure decryption algorithm
$I_e\,(J)$	Initial energy, value 2
$l_0\,(m)$	Threshold distance, value 87
E_{con} (nJ/bit)	Energy consumption, value 50
ε_{fs} (pJ/bit/m^2)	The energy required by the power amplification in the free space channel model, value 10
ε_{amp} (pJ/bit/m^4)	The energy required by the power amplification in the multipath fading channel model, value 0.0013
T_u	The neighbor node list of node u
hop_b	The number of hops from the base station to the current node
hop_s	The number of hops from the real source node to the current node
$\text{Minhops}_{i,b}$	The minimum number of hops that the node i is away from the base station
$\text{Minhops}_{i,s}$	The minimum number of hops that the node i is away from the real source node
E_i	Residual energy of node i

3.2.1. H-Hop Limited Flooding Starting from the Real Source Node. After the real source node detects that the object is nearby, it performs a limited flooding with a beacon message $SM = \{\text{ID}_s, \text{hop}_s\}$ [4], and the range of the flooding is limited within h hops. The SM contains the ID number of node ID_s that sent the message and the hop value from the real source node to the current node hop_s (the initial value is 0, plus 1 for each hop). When node u receives the beacon message SM from node v, if ID_v already exists in the neighbor node list T_u of node u, then the $\text{Minhops}_{v,s}$ of ID_v in T_u is updated with the smaller value hop_s in SM and the current $\text{Minhops}_{v,s}$ of node v. Otherwise, node u adds a new record to T_u, adds ID_v and hop_s into it, and hop_s at time $\text{Minhops}_{v,s}$.

Then we add 1 to hop_s and compare it with $\text{Minhops}_{u,s}$ for its own basic information, and update $\text{Minhops}_{u,s}$ with the smaller one as the current minimum number of hops from u to the real source node.

Node u replaces the ID in the message SM with its own ID, and forwards SM to its neighbor nodes together with the new hop_s. The neighbor nodes of u perform the same operation as u until hop_s count reaches h. At this point, h-hop limits the flooding process of the beacon message SM. Each node i within the range of h hops from the real source node knows the minimum number of hops $\text{Minhops}_{i,s}$ from itself to the real source node and the minimum number of hops from its neighbor nodes to the source node.

3.2.2. The Source Node Establishes Anonymous Proxy Paths. Definition. A is a node in the sensor network, the current node u selects v from its neighbor nodes as the forwarding node of the next hop. If the node v satisfies $\text{Minhops}_{v,A}$ -$\text{Minhops}_{u,A} \geq 1$, then we can say that the hop forwarding of the data packet from u to v is in a direction away from node A, where $A \neq u$ and $A \neq v$.

In addition, we define an optional set *u.gather* for each node u. The nodes in the set are the neighbors of u and satisfy the condition that the minimum number of hops from the source node is greater than the minimum number of hops from node u to the source node.

According to the energy requirements of receiving and sending data packets, the source node selects the neighbor nodes that meet the energy requirements according to the residual energy of each neighbor stored in its neighbor node list, and sends a detection data packet (h', Q) for possible proxy nodes. The detection data packet includes the number of hops h$'$ from the source node to the node (the initial value of h$'$ is 0) and a node queue Q; at the beginning, Q only contains the ID of the source node. Each time a detection data packet arrives at a node, the node adds its own ID to the node queue Q, at the same time the hop count h' is added 1.

We first select neighbor nodes that meet the energy requirements from its neighbor node list; then, we verify whether these selected neighbor nodes exist in their own optional set, and forward the detection data packet to the neighbor nodes in the optional set *u.gather*.

The neighbor node repeats this process until h' reaches h, and the detection process is completed. The node that receives the data packet at the h hop returns the queue Q to the real source node along the original path. Each node queue Q received by the real source node constitutes an anonymous proxy path. During the message forwarding phase, the anonymous proxy path is responsible for forwarding anonymous data packets to the proxy source node.

The detection process is over. At this time, the real source node has obtained several anonymous proxy paths.

3.2.3. Establishment of Candidate Region. For the anonymous proxy paths obtained in the previous section, the real source node chooses the first returned t as the candidate anonymous proxy paths. According to the predetermined upper and lower limits of the number of hops, we select all the nodes between the upper and the lower limits of the number of hops on the candidate anonymous proxy paths to form a candidate region.

For example, we suppose the real source node selects t node queues as follows: $Q_1, Q_2, \ldots, Q_t$. The region formed by all the nodes between the lth hop and the hth hop of each queue Q_i $(i = 1, 2, \ldots, t)$ is called the candidate region. At the same time, we call the region where the node from the first hop to the mth ($m = l{-}1$) hop of each queue Q_i ($i = 1, 2, \ldots, t$)is located as the visible region. The candidate region and the visible region in this example are shown in Figure 1.

3.2.4. Select Proxy Source Node. We select a node in the candidate region as the proxy source node of this communication. The path from the real source node to the proxy source node constitutes the anonymous proxy path of our anonymous communication.

In Figure 1, if the selected proxy source node Ps is the green node in the figure, then the anonymous proxy path $Rs \longrightarrow N_1 \longrightarrow N_2 \longrightarrow N_3 \longrightarrow N_4 \longrightarrow Ps$ from the real source node to the proxy source node is obtained.

3.3. Shortest Path Routing Algorithm Based on Residual Energy. The proxy source node uses the shortest path routing algorithm based on the residual energy to forward the data packets to the sink.

First, the proxy source node Ps obtains the residual energy of its neighbors by looking up the locally stored neighbor node list T_{Ps}, and selects all the neighbor nodes that meet the residual energy requirement, i.e., the residual energy can support the neighbor nodes that can receive and forward the data packets.

Then, the proxy source node searches its neighbor node list T_{Ps} again, selects a neighbor node with the smallest number of hops from the sink from the nodes that meet the remaining energy condition, and sends the data packet to the neighbor node.

Finally, after the neighbor node has received the data packet, it searches its neighbor node list in the same way as the proxy source node, selects the neighbor node that meets the energy requirements and has the smallest number of hops from the sink, which receives the data packet. This forwarding process is repeated till the data packet reaches the sink.

The shortest path routing algorithm based on the residual energy is shown below.

3.4. An Anonymous Communication Scheme Based on Proxy Source Node and Shortest Path Routing. Based on the previously proposed proxy source node selection mechanism, this section presents an anonymous communication scheme to protect the location privacy of the real source node. Our scheme is divided into three stages, namely, network initialization, anonymous path establishment, and anonymous message forwarding.

In the network initialization phase, the sink performs flooding of the beacon message BM in the network. At this stage, the nodes in the network can obtain the minimum number of hops from its own to the sink and the minimum number of hops from its neighbor nodes to the sink.

The anonymous path establishment phase includes four steps: the real source node performs h hop limited flooding, obtains an optional anonymous proxy path, establishes a candidate region, and selects the proxy source node. The real source node obtains the anonymous proxy paths from the source node to the proxy source node according to Section 3.2.

In the anonymous message forwarding phase, the source node first forwards the data packet from the real source node to the proxy source node via the anonymous proxy path obtained in Section 3.2, and then the proxy source node forwards the data packet to sink via the shortest path algorithm based on the residual energy to complete anonymous forwarding of the data packet.

3.4.1. Network Initialization Phase

① *Deployment of Sensor Networks*. In the sensor network, it includes a real source node, a sink node, and N ($N \in N^+$) wireless sensor nodes. These wireless sensor nodes communicate wirelessly with each other, and finally deliver the information to the sink node.

The topology of our wireless sensor network is shown in Figure 2, which depicts a random path from the real source node to the sink.

When the sensor network is deployed, each node u establishes a neighbor node list T_u. The neighbor node list contains the *ID* number $\mathrm{ID}_i$of neighbor node i, the minimum number of hops $\mathrm{Minhops}_{i,b}$ from neighbor node i to the base station, the minimum number of hops $\mathrm{Minhops}_{i,s}$ from neighbor node i to the real source node, and the residual energy value E_i of neighbor node i. The data structure of the neighbor node list is shown in Table 2. Among them, there is a situation where a neighbor node is of only $\mathrm{Minhops}_{i,b}$ but not $\mathrm{Minhops}_{i,s}$.

The neighbor node *ID* of each node u is obtained by the flooding process of the base station and the real source node. The minimum number of hops $\mathrm{Minhops}_{i,b}$ from the neighbor node i to the base station is obtained by the flooding process from the base station. The minimum number of hops $\mathrm{Minhops}_{i,s}$ from the neighbor node i to the real source node is obtained by the h hop limited flooding process starting from the source node.

Basic information of node u: each node u stores its own basic information through a quadruple (E_u, h, $\mathrm{Minhops}_{u,b}$, $\mathrm{Minhops}_{u,s}$). Among them, E_u represents the residual energy value of u. H-parameter is the number of the hops in the limited flooding performed by the real source node, which is initialized to null and is obtained during the flooding process of the base station. $\mathrm{Minhops}_{u,b}$ is the minimum number of the hops between node u and the sink node, $\mathrm{Minhops}_{u,s}$ is the minimum number of the hops between node u and the real source node. Both $\mathrm{Minhops}_{u,b}$ and $\mathrm{Minhops}_{u,s}$ are initialized to be the maximum number of the nodes in the network.

② *Flooding of Base Station*. The base station floods the beacon message $BM = \{\mathrm{ID}_b, \mathrm{hop}_b, h\}$ to the network, which contains the *ID* number of the node ID_b that sent the message, the hop value from the base station to the current node hop_b (the initial value is 0, plus 1 for each hop) and the number of hops h required for the establishment of the

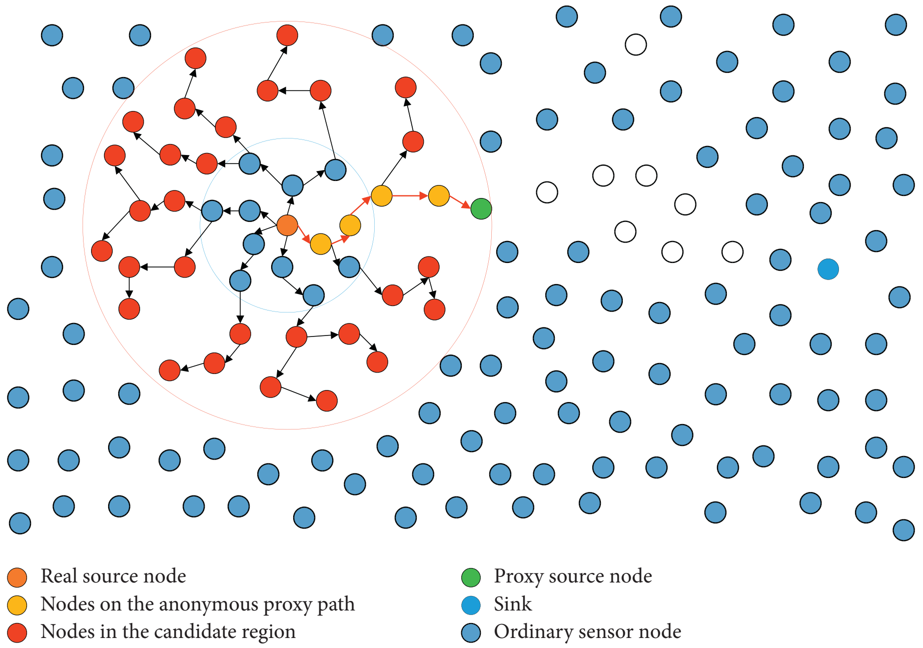

● Real source node
● Nodes on the anonymous proxy path
● Nodes in the candidate region
● Proxy source node
● Sink
● Ordinary sensor node

Figure 1: Establishment of candidate region.

```
cur_node $i$ = proxy source;
Initialize_neighbor_node_list($T_i$);
Initialize_packetInfo($pI$) = ($Q_{\text{proxy}}$, $E_{K_{s,b}}(M)$);
while(cur_node $i$ ! = sink) do
    $u$ = first_neighbor(node $i$);
    while($E_u \geq E_{\text{sender}}(n, l) + E_{\text{receiver}}(n)$) do
        save $\text{ID}_u$ in the array A[];
        $u$ = next_neighbor(node $i$);
    end while
    hops-min = $N$;
    for(node $u$ = first of (A[]); node $u$ in array A[]; $u$ = next of (A[]))
        if($\text{Minhops}_{u,b}$<hops-min)
        {
            hops-min = $\text{Minhops}_{u,b}$;
            $\text{ID}_{\text{hops-min}}$ = $\text{ID}_u$;
        }
    end for
    $i$ forwards the $pI$ to $\text{ID}_{\text{hops-min}}$;
    cur_node $i$ = $\text{ID}_{\text{hops-min}}$;
end while
```

Algorithm 1: Shortest path routing algorithm based on residual energy.

candidate region in the network. When node u receives the beacon message BM from node v, it performs the following operations in sequence:

First, u stores hops h in its own basic information.

Secondly, we search ID_v in the neighbor node list T_u. If the search is successful, i.e., there already exists node v in the list, we compare the newly received hop_b with the original minimum hop value $\text{Minhops}_{v,b}$, node u retains the smaller value and updates it as the minimum hop value $\text{Minhops}_{v,b}$ from u to the base station. Otherwise, we add a new record to T_u, assign the ID_v, and use the number of hops hop_b as the $\text{Minhops}_{v,b}$, i.e., the minimum number of the hops from v to the base station.

Thirdly, we add 1 to the number of hops hop_u received, compare it with $\text{Minhops}_{u,b}$ in the quadruple, and select the smaller value as the new $\text{Minhops}_{u,b}$.

Finally, we replace ID_v in the message with its own ID_u, together with the latest hop_b (at this time $\text{hop}_b = \text{hop}_b + 1$) and the received hop count h, construct a new beacon message $BM = \{\text{ID}_u, \text{hop}_b, h\}$ and forward it to the next node.

After flooding, each node i in the network is associated with h in the limited flooding, the minimum number of the

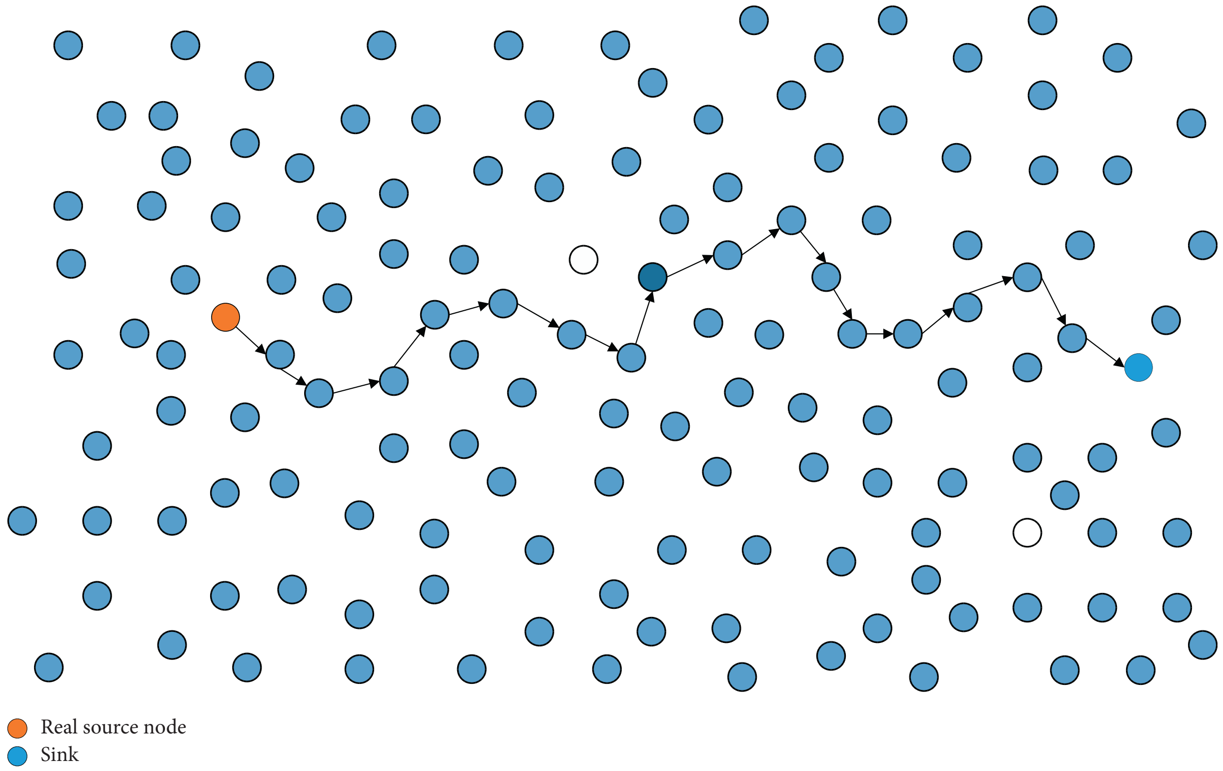


Figure 2: Network topology.

Table 2: Neighbor node list.

ID_i	$Minhops_{i,b}$	$Minhops_{i,s}$	E_i
ID_a	$Minhops_{a,b}$	$Minhops_{a,s}$	E_a
ID_c	$Minhops_{c,b}$	$Minhops_{c,s}$	E_c
ID_d	$Minhops_{d,b}$	$Minhops_{d,s}$	E_d
ID_e	$Minhops_{e,b}$	Null	E_e
. . .. . .			

hops from itself to the base station $Minhops_{i,b}$ and the minimum number of the hops from its neighbor nodes to the base station.

3.4.2. Anonymous Path Establishment Phase. The completely anonymous path refers to the anonymous path used for message forwarding from the real source node to the sink.

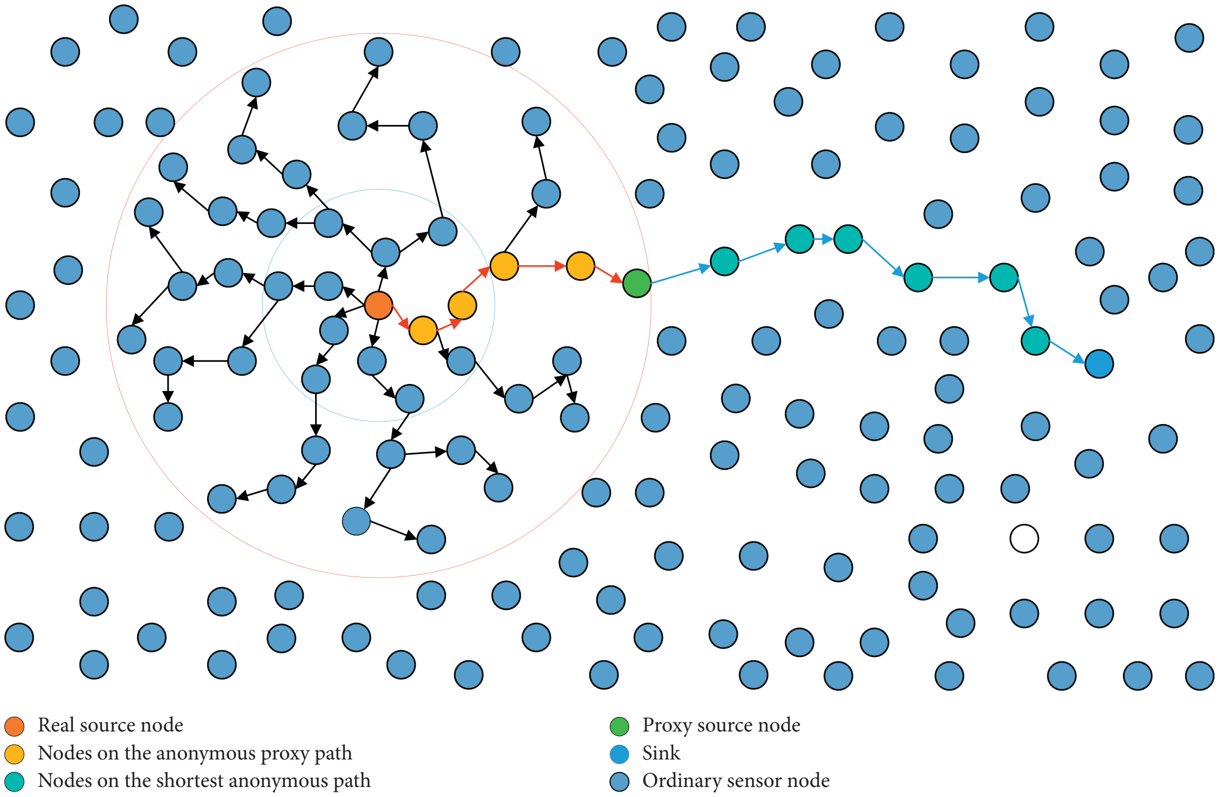
First of all, based on the content shown in Section 3.2, we establish a candidate region around the real source node. Before each communication, we will select a node in the candidate region as the proxy source node of the source node of this communication, and establish the first half anonymous proxy path from the real source node to the proxy source node. This process involves two stages in the proxy source node selection mechanism based on the candidate region: h hop flooding of the real source node and an optional anonymous proxy path. Secondly, based on the content shown in Section 3.3, we establish the second half of the shortest anonymous path from the proxy source node to the base station through the shortest path routing algorithm based on the residual energy.

The anonymous proxy path and the shortest anonymous path together constitute our anonymous communication path, as shown in Figure 3. The path from the real source node to the proxy source node is the anonymous proxy path, and the path from the proxy source node to the base station is the shortest anonymous path. We use the complete anonymous path to complete message forwarding.

3.4.3. Anonymous Message Forwarding Phase. This phase is divided into two stages. Stage 1: forward the data packet from the real source node to the selected proxy source node. Stage 2: the proxy source node forwards the data packet to the base station through the shortest path routing strategy based on the residual energy.

Stage 1: Rs-Ps message forwarding

According to the *ID* of the selected proxy source node, the real source node finds the queue where the proxy source



● Real source node
● Nodes on the anonymous proxy path
● Nodes on the shortest anonymous path
● Proxy source node
● Sink
● Ordinary sensor node

Figure 3: An anonymous path to protect the privacy of the source location.

node is located at Q_{proxy}, and then encrypts the message M to be sent with the *DH* key $K_{s,b}$ shared by the real source node and the base station, forming a data packet (Q_{proxy}, $E_{K_{s,b}}(M)$) with the queue Q_{proxy}. Send the data packet randomly to a certain number of neighbor nodes, and the selected neighbor nodes must include the first hop node in Q_{proxy}. For the neighbor nodes selected for each hop, there are two situations. First, if the node is not in the queue, it will randomly send the data packet to the next node. Second, if the node is in the queue, then it will select the next node based on the *ID* stored in Q_{proxy}.

Repeat this process till the proxy source node receives the data packet and stops the transmission. The process of Stage 1 is shown in Figure 4.

Stage 2: Ps-sink message forwarding

The proxy source node uses the shortest path routing strategy based on the residual energy mentioned in Section 3.3 to send the data packet to the base station. First, the proxy source node looks up the residual energy in the neighbor node list and finds the neighbor nodes that meet the energy requirements. We search the neighbor node list to find the one neighbor node that has the smallest minimum number of hops from it to the base station in these neighbor nodes that meets the energy requirements. The proxy source node sends the data packet to the neighbor node. Then the neighbor node forwards the data packet to the next node in the same way, until the base station receives the data packet and stops the transmission. After the sink receives the data packet, it can get the message by decrypting M with the operation of $D_{K_{s,b}}(M)$, where M is the message to be transmitted to the sink.

The process of Stage 2 is shown in Figure 5.

3.5. Enhanced Anonymous Communication. In order to further improve the anonymity of the scheme in this paper, an enhanced anonymous communication scheme is proposed in this section by dividing the candidate region into several sectors.

According to the scheme proposed above, if the real source node selects the proxy source node before data transmission, it selects the node on the same or similar anonymous proxy path several times in a row. The network will be affected by the node receiving and forwarding the data. The generated traffic will be concentrated in a certain area for a period of time, which will make it easy for the adversary to guess the location of the real source node through traffic analysis. Therefore, in order to make the real source node evenly select a node on each anonymous proxy path and further resist traffic analysis attacks, we propose an enhanced anonymous communication strategy based on sector division.

For the candidate region established in Section 3.2, we define $R_{\min}$ to represent the distance between the selected lower limit and the real source node, and define $R_{\max}$ to represent the distance between the selected upper limit and the real source node. Then, we divide the candidate region into several equal sectors, each sector spans an angle μ, the total number of the sectors is $s = (2\pi/\mu)$, and we define these

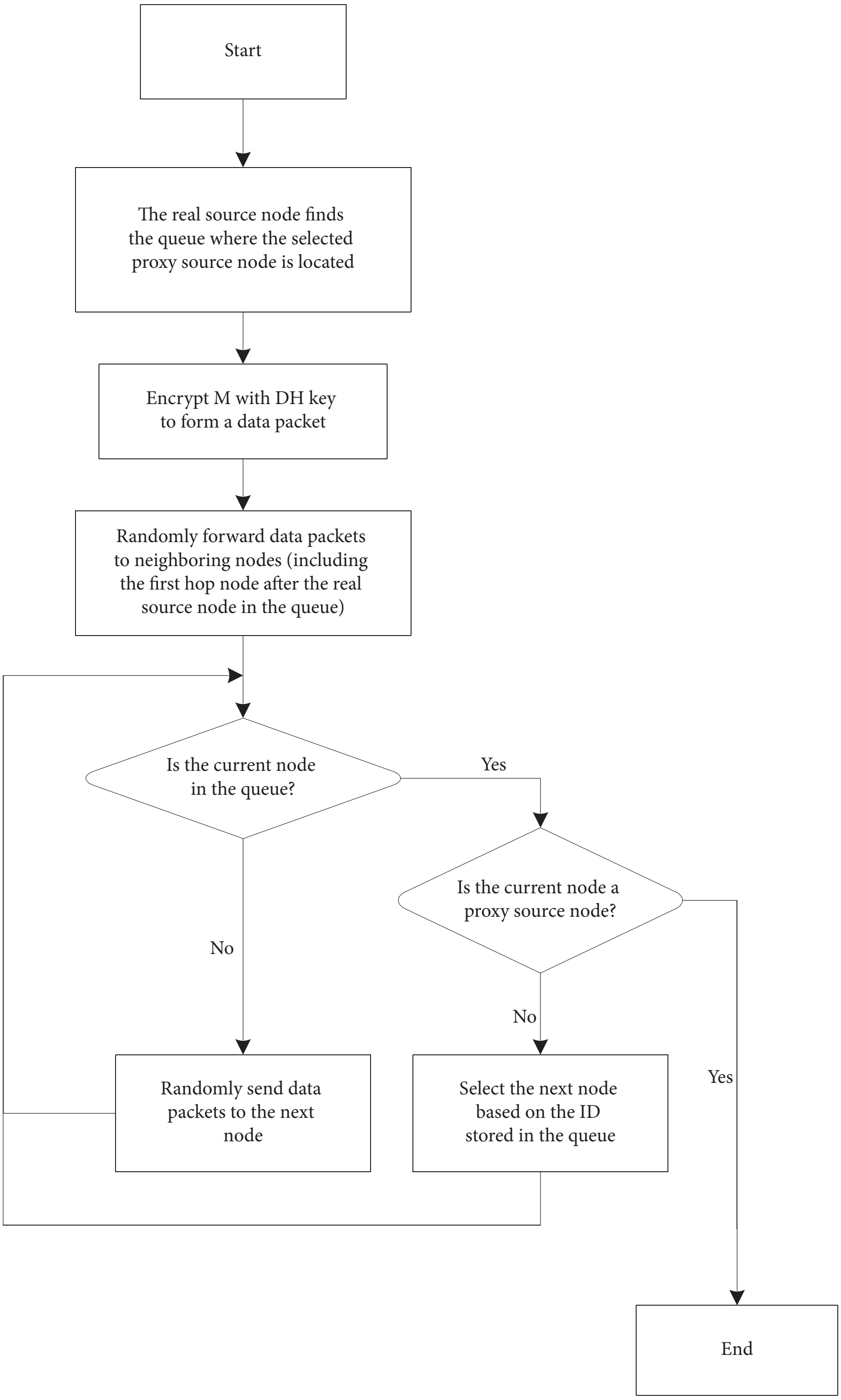

Start
The real source node finds the queue where the selected proxy source node is located
Encrypt M with DH key to form a data packet
Randomly forward data packets to neighboring nodes (including the first hop node after the real source node in the queue)
Is the current node in the queue?
Yes
No
Is the current node a proxy source node?
No
Yes
Randomly send data packets to the next node
Select the next node based on the ID stored in the queue
End

Figure 4: Forward the data packet from RS to PS.

sectors as area$_1$, area$_2$, , area$_\mu$. We can use the following equation to calculate μ.

$$\mu = \frac{\arcsin(R_{\min}/R_{\max}) + \arcsin(R_{\min}/H)}{\pi}, \qquad (4)$$

which is used to calculate the value of the angle μ [28]. Among them, $R_{\min}$ is the radius of the visible region, $R_{\max}$ is the radius of the candidate region, and H is the number of the hops from the real source node to the sink. Then we can

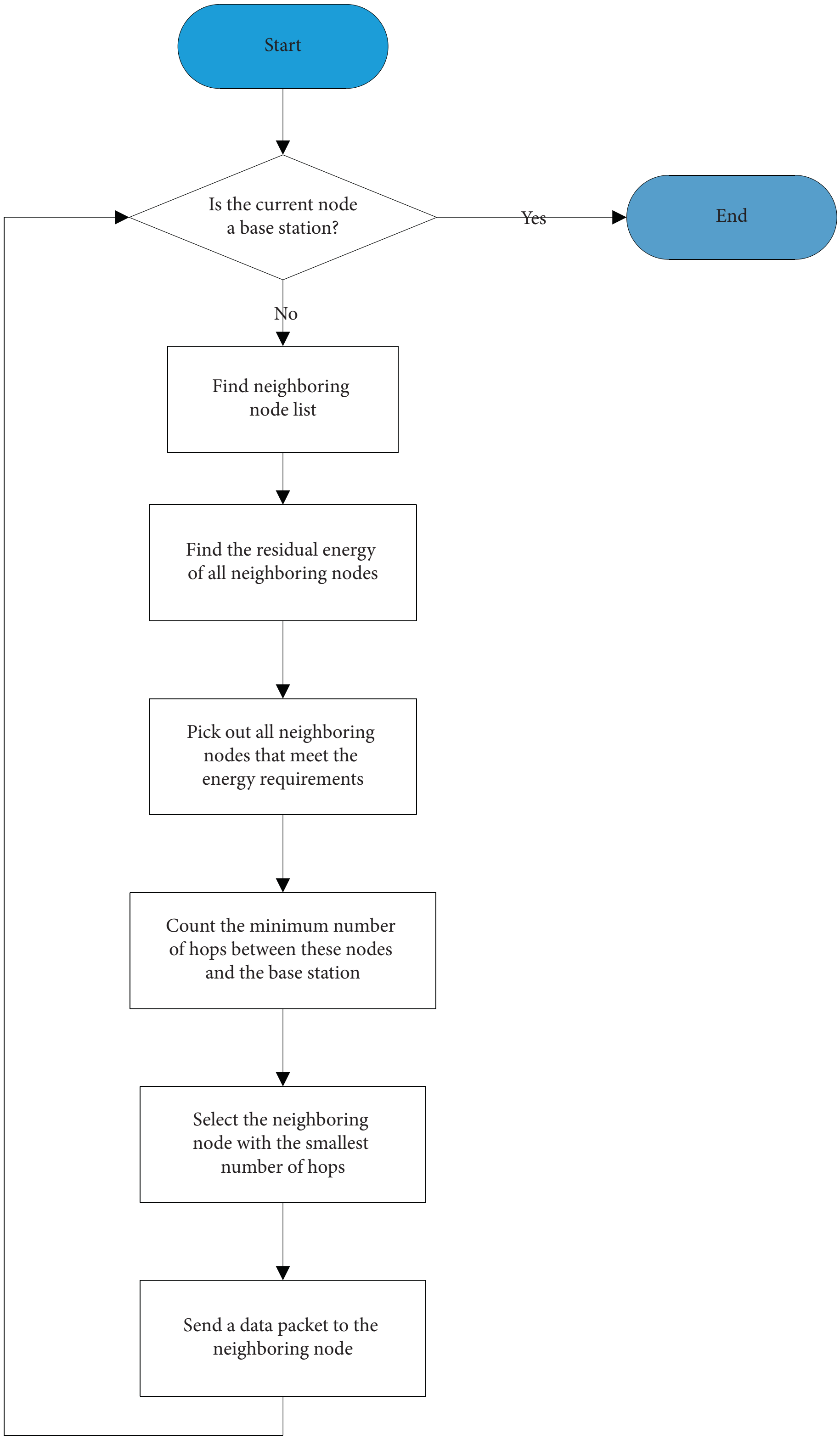

Start
Is the current node a base station?
Yes
End
No
Find neighboring node list
Find the residual energy of all neighboring nodes
Pick out all neighboring nodes that meet the energy requirements
Count the minimum number of hops between these nodes and the base station
Select the neighboring node with the smallest number of hops
Send a data packet to the neighboring node

Figure 5: Forward the data packet from PS to the sink.

calculate the total number of sectors s via μ. The candidate region divided into sectors is shown in Figure 6.

When selecting the proxy source node, we first randomly select an area area$_i$, where i falls in $[1, \mu]$. Then, we generate a random angle β, which is in the range of $[(i-1)\mu, i\mu]$. Finally, we generate a random distance d, which satisfies the range $[R_{\min}, R_{\max}]$. The relative position of the selected proxy source node is $(x\ d + d\ cos(\beta), y\ d + d\ sin(\beta))$, where (x, y) are the coordinates of the real source node. Because the location of the proxy source node is randomly selected, we may not

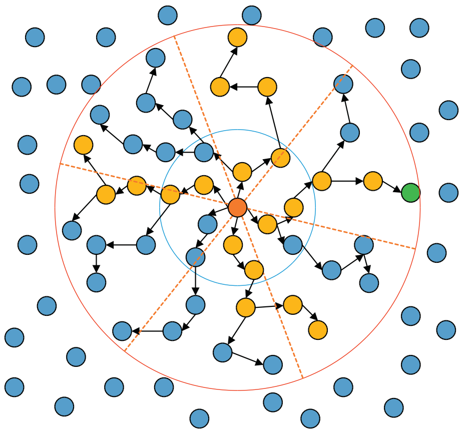


Figure 6: Sector division in the candidate area. The angle of each sector is μ, and there are (s) sectors in total.

see any node in the desired region. If there is no node in the desired region, the last hop node routed to the selected location path becomes the proxy source node.

It is important to note that there may be duplicate nodes in these paths, but this will not affect our operations. When we select the proxy source node through the candidate region before each data transmission, it should alternately select the proxy source node from different sectors instead of the same sector. That is, when the real source node selects the node in the area area$_i$ (i = 1, 2, , μ) as the proxy source node in a packet transmission, it will not select nodes in the adjacent area of area$_i$ used as the proxy source node in the next packet transmission, and the node in the area$_i$ will not be selected as the proxy source node in the subsequent k (k≤(μ/4)) data packet transmissions.

In this way, the traffic in the network is evenly distributed in different areas within a period of time, instead of being concentrated in the same area, which makes it difficult for the adversary to track and guess the location of the real source node.

4. Results and Discussion

In this paper, we have proposed an anonymous communication scheme based on the proxy source node and the shortest path routing algorithm to protect source location privacy in the IoT. The scheme has the location privacy of the source node, anonymity, and can also prevent adversaries from collecting and analyzing communication messages in the whole network, and monitoring of network traffic in a certain region, such as impersonation and backtracking attacks. Table 3 compares the security performance of our scheme with Random Walk, GROW [35, 36], and ARPLP scheme.

4.1. Source Node Privacy. The scheme in this paper selects the proxy source node to replace the real source node to send data packets, and there is no direct connection between the proxy source node and the real source node. And the proxy source node is randomly selected from the candidate region, that is to say, there is also no direct connection between the selected proxy source nodes each time. Therefore, for the adversary, they do not know any strategy about how the proxy source node is selected, and will not obtain the location information of the real source node through the proxy source node. Specifically, assume that two consecutive data transmissions use x and y as the proxy source nodes; however, because they are randomly selected, there is no connection between x and y. The adversary cannot judge the selection rule of the proxy source node through the two selections of the proxy source node.

At the same time, this scheme ensures that the selected proxy source node is far enough from the real source node through the selection mechanism of the proxy source node within the candidate region, thereby increasing the distance between the real source node and the proxy source node, and making the real source node have better privacy. In the process of establishing anonymous proxy paths, since the detection data packet is sent to the nodes that meet the energy requirements and are in the optional set, the number of hops from the next node to the real source node is greater than the number of hops from the current node to the real source node. In this way, after h hops, the probability that the proxy source node is less than $h/5$ hops from the real source node will be greatly reduced from the original $P = 1 - e^{-(h/25)}$.

In conclusion, this scheme can well hide the location of the real source node and protect the location privacy of the source node.

4.2. Anonymity. The scheme in this paper realizes the anonymity of the transmitted messages and the anonymity of the nodes in the wireless sensor networks.

Before the data packet transmission starts, the real source node and the base station jointly negotiate a *DH* shared key $K_{s,b}$, and only the real source node and the base station can know this key. The data to be transmitted are encrypted with this shared key, which forms the data packet with Q_i and is forwarded to the next hop. Passing through the proxy source node, the data arrive at the base station, and the base station uses the shared key to decrypt the data packet. When the data packet is transmitted from the real

Table 3: Comparison of security performance.

Scheme	Source location privacy	Anonymity	Anti-Impersonation	Backtracking attack
Random walk	No	No	No	No
GROW	Yes	No	No	Yes
ARPLP	Yes	Yes	No	Yes
Proposed	Yes	Yes	Yes	Yes

source node to the base station through the anonymous path we have established, each intermediate node does not know the shared key; therefore, others cannot decrypt the data packet and tamper with the content of the data packet, and the packet does not contain any information about the identity of the node.

At the same time, anonymity also includes the anonymity of the source node in the network and the anonymity of the communication relationship. The source node sends the information to the proxy source node through the anonymous proxy path, and then the proxy source node sends the information to the base station through the shortest path routing based on the residual energy. There is no identity information involved in the process. Each node only knows who its previous hop and next hop are, and does not know the source and destination of the information.

4.3. Anti-Impersonation Attack. Impersonation attacks refer to malicious nodes pretending to be legitimate nodes to forward messages, causing messages to be tampered with or interrupted in forwarding. In our scheme, it is not feasible for a node in the network to pretend to be a proxy source node. The reason is as follows:

When the real source node sends the data packet to the next node, the selected neighbor nodes must include the first hop after the source node *ID* recorded in the selected anonymous proxy path, and then the current node also selects the next node according to the *ID* recorded in the path. If the malicious node is not in the selected anonymous proxy path, then its *ID* will not appear in the selected anonymous proxy path; if the malicious node is in the selected initial path, but the real source node knows the *ID* of the selected proxy source node, it will not select the nodes with other hops on the path.

Specifically, if the real source node selects node p in the candidate region as the proxy source node before a message transmission starts. The node p is a node on the anonymous proxy path Q_r. Then, in the first stage of the anonymous message forwarding, the real source node forwards the data packet $(Q_r, E_{K_{s,b}}(M))$ to a certain number of the neighbor nodes, containing the node represented by the first *ID* other than the real source node *ID* recorded in Q_r. If the malicious node is not in the selected Q_r, then its *ID* will not appear in Q_r and will not affect the transmission of the data packets; if the malicious node is in Q_r, since the real source node knows ID_p on Q_r, it will not select the nodes with other hops on Q_r, but will stop until the data packet is transmitted to p.

4.4. Backtracking Attack. In wireless sensor networks, backtracking attack means that the adversary located near the base station will observe that the destination node receives the data information, and then start from the destination node and trace back hop-by-hop along the path until the source sensor node is found, which is the sender of the information. Our anonymous communication scheme can resist adversary backtracking attacks. The reason is as follows:

In our scheme, a candidate region formed by a flooding mechanism is set up, in which all nodes may become proxy source nodes and send messages instead of the real source nodes. In the process of message forwarding from the real source node to the proxy source node, there will be a lot of branch traffic to confuse the adversary, so the adversary can only trace back to the proxy source node, but cannot continue to trace back to find the location of the real source node.

We used PyCharm [37] to simulate the proposed scheme. For the energy consumption model presented in Section 3.1, the simulation results are as follows: Figure 7 shows the change trend of the energy consumed by the sender as the distance between the sender and the receiver changes when the number of the message bits transmitted are 50, 100, 150, and 200, respectively. We can see that as the distance between the sender and the receiver increases, the energy consumed by the sender, i.e., E_{sender}, is also increasing. The larger the number of bits, the more energy is consumed. Figure 8 shows the change trend of the energy consumed by the sender as the number of the message bits transmitted changes when the distance between the sender and the receiver is 50, 100, 150, and 200, respectively. It can be seen that as the number of message bits continues to increase, the energy consumed by the sender is also increasing. The greater the distance, the more energy is consumed.

Figure 9 shows the change trend of the energy consumed by the sender when the number of transmitted bits and the distance between the sender and the receiver simultaneously vary from 0 to 200.

At the same time, we simulated the relationship between the number of the transmitted message bits and the energy

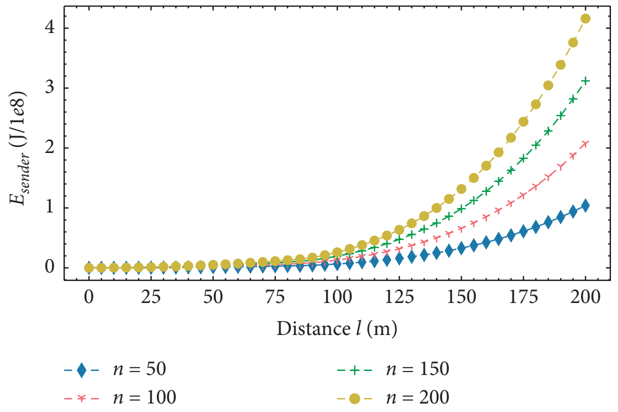


FIGURE 7: The energy consumption of the sender varies with the transmission distance.

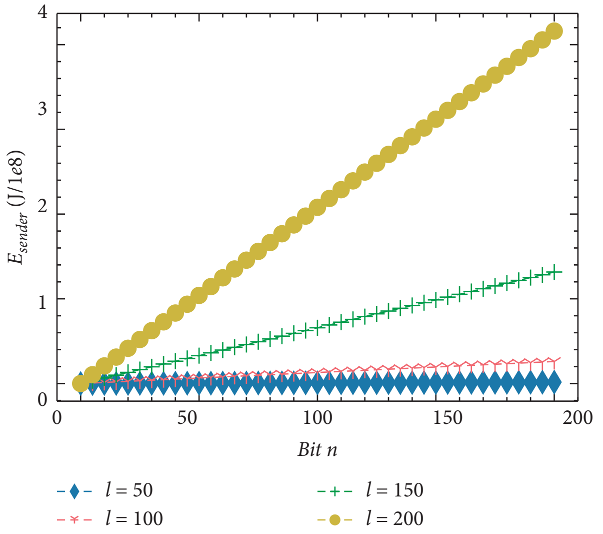


FIGURE 8: The energy consumption of the sender varies with the number of transmission bits.

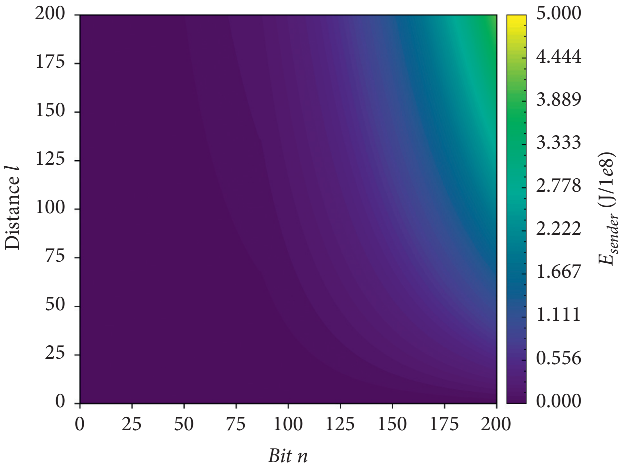


FIGURE 9: The energy consumption of the sender varies with the transmission distance and the number of transmission bits.

consumed by the receiver in a segmented form. As shown in Figure 10, the number of the transmitted message bits is divided into four closed intervals, which are [0, 50], [51, 100], [101, 150], and [151, 200], and the energy consumed by the receiver in the four intervals is obtained. In each interval, as the number of message bits increases, the energy

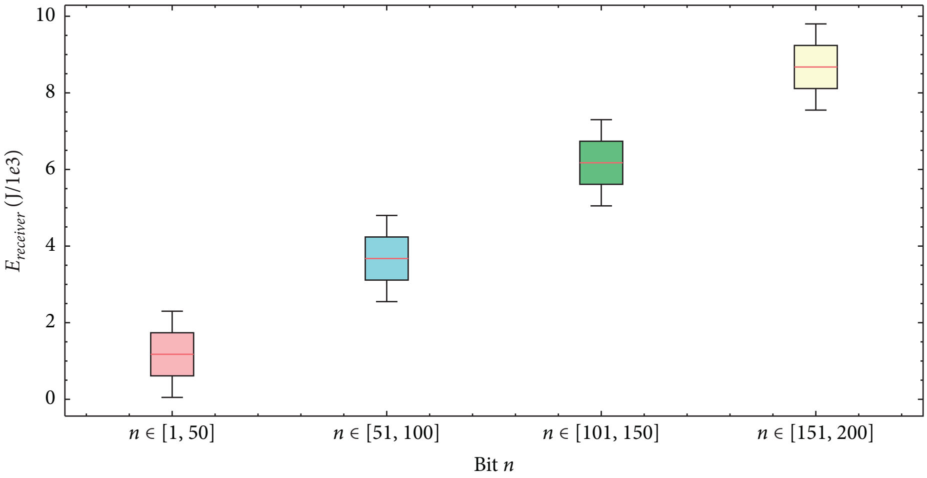


Figure 10: Energy consumption diagram of received data.

consumed by the receiver will increase geometrically. At the same time, as the number of bits in each interval increases, the energy consumed by the receiver will also increase linearly.

5. Conclusion

In the context of the Internet of Things, while wireless sensor networks are widely used in various fields, they also face many security problems. Among them, the privacy protection of the source location is a very important security issue. In response to this problem, we proposed an anonymous communication scheme that protects the privacy of the source location in the IoT. By establishing a candidate region, the proxy source node is randomly selected in the candidate region to replace the real source node to send data packets, thereby achieving the purpose of protecting the location of the real source node. In the process of data packet transmission from the proxy source node to the sink, we used the shortest path routing algorithm based on the residual energy, so as to achieve the goal of saving energy and improving efficiency. But the work of this article does not involve the part of protecting the base station, i.e., all the nodes in the network know the location of the base station, and there is no specific application part. Therefore, our future work will focus on how to protect the location privacy of the base station while ensuring the privacy of the overhead and source location, and actively integrate it with practical applications.

Data Availability

No data were used to support this study.

Conflicts of Interest

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

Acknowledgments

This study was partly funded by EU Horizon 2020 DOMINOES Project (Grant Number: 771066).

References

[1] N. Papernot, P. McDaniel, and A. Sinha, “Towards the science of security and privacy in machine learning,” 2016, https://arxiv.org/abs/1611.03814.

[2] Y. Wang, G. Yang, and T. Li, “Belief and fairness: a secure two-party protocol toward the view of entropy for IoT devices,” *Journal of Network and Computer Applications*, vol. 161, Article ID 102641, 2020.

[3] M. Lu, Z. Zhao, and X. Tang, “Research on source location privacy protection based on phantom routing,” *Information Security and Technology*, vol. 3, no. 10, pp. 72–76, 2012.

[4] J. Chen, B.-X. Fang, L.-H. Yin, and S. Su, “A source-location privacy preservation protocol in wireless sensor networks using source-based restricted flooding,” *Chinese Journal of Computers*, vol. 33, no. 9, pp. 1736–1747, 2010.

[5] P. Mohassel and Y. Zhang, “SecureML: a system for scalable privacy-preserving machine learning,” in *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*, pp. 19–38, IEEE, San Jose, CA, USA, May 2017.

[6] K. Bonawitz, V. Ivanov, and B. Kreuter, “Practical secure aggregation for privacy-preserving machine learning,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, IEEE, Dallas, TX, USA, October 2017.

[7] F. Li, D. Wang, and Yi. Wang, “Blockchain-based trust management in distributed internet of things,” *Wireless Communications and Mobile Computing*, vol. 10, no. 1155, Article ID 8864533, 2020.

[8] Y. Wang, Y. Wang, and Z. Wang, “Research cooperations of blockchain: toward the view of complexity network,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 92, pp. 1–14, 2020.

[9] Y. Wang, G. Yang, A. Bracciali et al., “Incentive compatible and anti-compounding of wealth in proof-of-stake,” *Information Sciences*, vol. 530, pp. 85–94, 2020.

[10] T. Li, Y. Chen, and Y. Wang, “Rational protocols and attacks in blockchain system,” *Security and Communication Networks*, vol. 10, no. 1155, Article ID 8839047, 2020.

[11] F. Li, R. Ge, and Y. Wang, “Tesia: a trusted efficient service evaluation model in IoT based on improved aggregation signature,” *Concurrency and Computation: Practice and Experience*, vol. 10, no. 1002, 2020.

[12] C. Zhao, S. Zhao, and M. Zhao, “Secure multi-party computation: theory, practice and applications,” *Information Sciences*, vol. 476, pp. 357–372, 2019.

[13] H. Wang, G. Han, and C. Zhu, “TCSLP: a trace cost based source location privacy protection scheme in wsns for smart cities,” *Future Generation Computer Systems*, vol. 101, pp. 965–974, 2020.

[14] X. Zhu, “Research on privacy protection strategy based on source node location in wireless sensor network,” *Communication Power Technology*, vol. 37, no. 2, pp. 42-43, 2020.

[15] G. Han, M. Xu, and Y. He, “A dynamic ring-based routing scheme for source location privacy in wireless sensor networks,” *Information Sciences*, vol. 504, 2019.

[16] S. D. Muruganathan, D. C. F. Ma, and R. I. Bhasin, “A centralized energy-efficient routing protocol for wireless sensor networks,” *IEEE Communications Magazine*, vol. 43, no. 3, pp. S8–S13, 2005.

[17] L. Mutalemwa and S. Shin, “Achieving source location privacy protection in monitoring wireless sensor networks through proxy node routing,” *Sensors (Basel, Switzerland)*, vol. 19, no. 5, 2019.

[18] B. Chakraborty and S. Verma, *Differentially Private Location Privacy Preservation in Wireless Sensor Networks*, Vol. 104, Springer US, New York, NY, USA, 2019.

[19] J. Wang, R. Zhu, and S. Liu, “Node location privacy protection based on differentially private grids in industrial wireless sensor networks,” *Sensors*, vol. 18, no. 2, 2018.

[20] P. Spachos and D. Toumpakaris, “Angle-based dynamic routing scheme for source location privacy in wireless sensor networks,” in *Proceedings of the 79th Vehicular Technology Conference*, IEEE, Seoul, Korea, May 2014.

[21] Y. Liu and Y. Xu, “Source location privacy protection scheme based on variable included angle dynamic routing in WSNs,” *Computer Application Research*, vol. 35, no. 1, pp. 257–260, 2018.

[22] L. Bai, L. Li, and S. Qian, “Privacy protection algorithm based on expected phantom source node in wireless sensor network,” in *2016 7th IEEE International Conference on Software Engineering and Service Science*, pp. 1006–1009, IEEE, Beijing, China, August 2016.

[23] S. Li, Y. Xiao, L. Q, and Z. Qi, “A novel routing strategy to provide source location privacy in wireless sensor networks,” *Wuhan University Journal of Natural Sciences*, vol. 21, no. 4, pp. 298–306, 2016.

[24] L. Lin, *Research on Privacy Protection of Node Location and Data in Wireless Sensor Network*, Beijing University of Posts and Telecommunications, Beijing, China, 2015.

[25] F. Li, C. Cui, and D. Wang, “Privacy-aware secure anonymous communication protocol in CPSS cloud computing,” *IEEE Access*, vol. 10, no. 1109, Article ID 2982961, 2020.

[26] S. Sharma and K. Ghosh, “Providing Privacy and security of Wireless sensor network using ACTOR nodes,” *Control Theory and Informatics*, vol. 4, no. 4, 2014.

[27] G. Tan, *Research on Privacy Protection of Source Node Location in Wireless Sensor Network*, Anhui University, Hefei, China, 2014.

[28] Z. Zhao, Y. Liu, and F. Zhang, “Research on WSN source location privacy preserving routing based on angle and probability,” *Journal of Shandong University (Science Edition)*, vol. 48, no. 9, pp. 1–9, 2013.

[29] J. P. Sheu and J. R. Jiang, “Anonymous path routing in wireless sensor networks,” in *Proceedings of the IEEE International Conference on Communications*, pp. 634–640, IEEE, Beijing, China, May 2008.

[30] Y. Li and J. Ren, “Source-location privacy through dynamic routing in wireless sensor networks,” in *2010 Proceedings IEEE INFOCOM*, IEEE, San Diego, CA, USA, March 2010.

[31] J. Ren, Y. Zhang, and K. Liu, “An energy-efficient cyclic diversionary routing strategy against global eavesdroppers in wireless sensor networks,” *International Journal of Distributed Sensor Networks*, vol. 151, 2013.

[32] L. Zhou and Y. Shan, “An anonymous routing scheme for preserving location privacy in wireless sensor networks,” in *Proceedings of the IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 10, no. 1109, pp. 262–265, IEEE, Chengdu, China, March 2019, Article ID 8728980.

[33] F. Li, Z. Liu, and L. Li, “Privacy-aware PKI model with strong forward security,” *International Journal of Intelligent Systems*, vol. 10, no. 1002, p. 22283, 2020.

[34] C. Cui, F. Li, and T. Li, *Research on Direct Anonymous Attestation Mechanism in Enterprise Information Management Enterprise Information Systems*, vol. 55, pp. 1–17, 2019.

[35] X. Yong and L. Schwiebert, “Preserving source location privacy in monitoring-based wireless sensor networks,” in *Proceeding of 20th International Conference on Parallel and Distributed Processing Symposium (IPDPS’06)*, Island of Rhodes, Greece, January 2006.

[36] L. Zhou and Q. Wen, “Providing location privacy against a global adversary in wireless sensor networks,” *Journal of Information and Computational Science*, vol. 10, no. 15, pp. 5043–5053, 2013.

[37] Q. Hu and L. Ma, “Deepgraph: a pycharm tool for visualizing and understanding deep learning models,” in *Proceedings of the 2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, IEEE, Nara, Japan, December 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6695304, 7 pages
https://doi.org/10.1155/2021/6695304

Research Article

Two-Party Secure Computation for Any Polynomial Function on Ciphertexts under Different Secret Keys

Bingbing Jiang

Computer Science and Technology Department, Nanjing University, Nanjing, China

Correspondence should be addressed to Bingbing Jiang; njubing.jiang@gmail.com

Received 8 December 2020; Revised 8 January 2021; Accepted 8 February 2021; Published 22 February 2021

Academic Editor: Jinguang Han

Copyright © 2021 Bingbing Jiang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Multikey fully homomorphic encryption proposed by Lopez-Alt et al. (STOC12) is a significant primitive that allows one to perform computation on the ciphertexts encrypted by multiple different keys independently. Then, several schemes were constructed based on decisional small polynomial ratio or learning with errors. These schemes all require an expansion algorithm to transform a ciphertext under a single key into an encryption of the same message under a set of keys. To achieve the expansion algorithm without interaction with these key-keepers, their encryption algorithm not only outputs a ciphertext of a plaintext but also exports auxiliary information generated from the randomness used in the former encryption process. Beyond that, the size of the ciphertext encrypted by multiple keys increases linearly or quadratically in the number of participants. In this paper, we studied the problem whether someone can directly perform arbitrary computation on ciphertexts encrypted by different keys without any auxiliary information in the output of the encryption algorithm and an increase in the size of the ciphertext in the expansion algorithm. To this end, we proposed a novel and simple scheme of secure computation on ciphertexts under two different keys directly without any auxiliary information. In other words, each party just provides its own ciphertexts encrypted by the GSW scheme (CRYPTO13). In the procedure of executing evaluation on these ciphertexts, the size of the new ciphertext remains the same as that of the GSW ciphertext.

1. Introduction

The concept of multikey fully homomorphic encryption was proposed by Lopez-Alt et al. [1], which allows someone to perform arbitrary computations on the ciphertexts encrypted by multiple different secret keys. Specifically, each party independently encrypts input x_i, to obtain a ciphertext $c_i = \mathrm{Enc}_{pk_i}(x_i)$, and one can homomorphically evaluate an arbitrary function on these encrypted data without interaction between them. After this, there has been a lot of research [2–12] for its assumptions, functionalities, and performance.

The main application of multikey FHE is that a plurality of parties is informed to engage in a computing task after they have submitted their data. This is a significant difference from the applications of the traditional (single-key) encryption schemes. For example, two hospitals want to cooperate and study the influence factors of some disease. However, the data of these patients has been encrypted and stored in their own servers ahead of this cooperation. How could an evaluation algorithm be performed directly on these ciphertexts without decrypting them? In [1], Lopez-Alt et al. focused on a problem whereby a (untrusted) cloud server wants to perform some computations over data from multiple clients without interacting with them after each client transmits their own (encrypted) input to the cloud and other clients. In the scheme proposed by Lopez-Alt et al. [1], although a ciphertext only contains an encryption of a plaintext, the size of a ciphertext under multiple secret keys becomes much larger than that of the original ciphertext and its security is based on the nonstandard assumption. The ciphertext's length is related to the number of participants where the former increases at least linearly in the later. In the scheme of Clear and McGoldrick [3], an encryption of a message contains a universal mask U generated by another public-key encryption scheme. Also, the ratio of the size of the ciphertext under multiple keys and that under single key grow quadratically with an increase in the number of the associated participants. Afterwards, Mukherjee and Wichs [2] proposed an optimized scheme with a simple generation of the universal mask. However, there is still auxiliary information in the encryption algorithm and the ratio remains quadratic. Following the previous works, there are two independent researches about multikey fully homomorphic encryption introduced by Brakerski and Perlman [5] and Peikert and Shiehian [4], respectively. In the former scheme, although the authors replaced the algorithm of the universal mask with the bootstrapping algorithm, the ciphertext's growth rate was still linear and their evaluation keys were generated by the previous multikey fully homomorphic encryption schemes. There are two versions in the paper in [4]. In the first scheme, the encryption of a message contains a commitment of the message and an encryption of the randomness used in the former commitment algorithm. The ratio becomes linear. In the second one, the encryption algorithm only outputs a ciphertext of a message, but the ratio becomes quadratic and the evaluation keys are generated by the first scheme. In [13], the growth rate is quadratic, and the output of the encryption algorithm also contains auxiliary information except a ciphertext of a plaintext. Recently, Chen et al. [6] proposed a multikey FHE scheme based on the ring-LWE (Learning with Errors) assumption, in which their ciphertext-extension algorithm only generates the evaluated keys for the scheme with multiple keys but the size of the ciphertext under multiple keys also relies on the number of associated parties.

The first multikey fully homomorphic encryption was proposed by Lopez-Alt et al., but their solution is based on nonstandard assumptions. Subsequent solutions, despite being based on standard cryptographic assumptions (LWE), have two common shortcomings. The first shortcoming is that they require the encryption of not only the plaintext but also random numbers that have been used; namely, $c = \mathrm{Enc}(\mathrm{pk}, m, r)$, and $U = \mathrm{Enc}(\mathrm{pk}, r)$. Each ciphertext must be attached with additional information U. The second one is that the length of the ciphertext increases linearly or quadratically with the number of participants. In this paper, our main research problem is how to directly perform secure computation on ciphertext data c directly provided by each user without any additional information U. These ciphertext data are encrypted with different secret keys. Our main focus here is the case of encryption with two different keys. We begin by taking the GSW13 encryption scheme [14] into consideration as we notice that the main process of its decryption algorithm is the inner product of two vectors; that is, $\langle c, v\rangle = m\ d + e$, where d is a large constant. As such, if we want to calculate the product of ciphertexts c_1 and c_2 encrypted with different secret keys, we only need to calculate $c_1 \cdot c_2^T$. This is because $v_1^T \cdot c_1 \cdot c_2^T \cdot v_2 = (m_1 d + e_1)(m_2 d + e_2) = m_1 m_2 d^2 + d(m_1 e_2 + m_2 e_1) + e_1 e_2$. The final result is desirable, with $m_1 m_2$ being one of its factors. However, there is another problem: the constant factor becomes d^2, and small noises e_1 and e_2 are also multiplied by a large number. Therefore, we must find a way to decrease the constant factor to d, while keeping the noises within an acceptable range. Because the noise in the ciphertext grows with an increase in the number of addition and multiplication operations, when it increased to some value defined by the public parameters, it may cause incorrect decryption of the output ciphertext. Therefore, we should reduce the noise growth in evaluation.

Our approach is to decrypt $c_1 \cdot c_2$ in two steps without directly multiplying it by two secret keys. Instead, a single secret key is first used to decrypt it, that is, $v_1^T \cdot c_1 \cdot c_2^T = (m_1 d + e_1) c_2$ (denoted as tc_1), before tc_1/d is calculated and rounded to obtain $tc = m_1 c_2$. Finally, another secret key is used to decrypt tc for the final plaintext $m_1 m_2$. During the process, noises have been kept at a low level without being multiplied by a large constant factor. To sum up, the above description explains how to perform the multiplication operation on ciphertexts encrypted with two different keys. The addition operation can be transformed to the multiplication operation; that is, $c_1 + c_2 = (c_1 \cdot c_2') + (c_1' \cdot c_2)$, where c_1' and c_2' are encrypted from plaintext 1 with different secret keys. Till this step, we completed the addition and multiplication operations on ciphertexts encrypted with two different secret keys. However, this scheme has a shortcoming: the multiplication operation can only be performed once as the result of the multiplication operation on the ciphertexts encrypted with two different secret keys cannot be multiplied by other ciphertexts. In order to enable the support of polynomial calculation, we can write any polynomial f with $u + v$ inputs $(x_1, \ldots, x_u, y_1, \ldots, y_v)$ as follows: $f = \sum_{i=1}^{w} (f_i \cdot g_i)$, where the inputs of f_i are $x_1, \ldots, x_u$, and the inputs of g_i are $y_1, \ldots, y_v$. In this way, we can first use the single-key fully homomorphic encryption scheme to calculate f_i and g_i to obtain intermediate results and then calculate the final results with our proposed method. Therefore, our secure computation only involves the GSW13 encryption scheme without the requirement for additional information U. Moreover, unlike previous schemes where a ciphertext's size grows linearly or quadratically as the number of secret keys increases, the ciphertext in our scheme always maintains its original size.

Our Contributions. We proposed a protocol that allows one to perform any polynomial functions on the GSW ciphertexts under two different keys directly. Unlike the previous works, each party just provides the GSW ciphertexts without anything auxiliary of the private inputs and the size of the new ciphertext remains invariant when executing evaluations on these ciphertexts. In our *Addition* and *Multiplication* algorithms on ciphertexts under two different keys, the noise increases linearly. Compared to the scheme in [1], our scheme is based on the standard assumption. Our scheme reduces the size of the ciphertext under a single key from $\mathcal{O}(n^4 \log^4 q)$ in [2, 3] to $\mathcal{O}(n^2 \log^2 q)$, where n is the lattice dimension and q is a modulus. Compared to the scheme in [5], our scheme does not require the expensive technique of bootstrapping to transform a ciphertext under a single key to a ciphertext under a set of keys. In the first scheme of [4], the size of the ciphertext under a single key is $\mathcal{O}(n^3 \log^3 q)$. The second scheme of [4] requires its first scheme to generate a public key with larger size. Different from the scheme in [6], the size of the public key in our scheme is the same as that of the GSW13 scheme, whereas it is $\mathcal{O}(\log q)$ times the size of the GSW13 scheme.

2. Related Work

In the scheme proposed by Lopez-Alt et al. [1], although a ciphertext only contains an encryption of a plaintext, the size of a ciphertext under multiple secret keys becomes much larger than that of an original ciphertext and their security is based on the nonstandard assumption. The ciphertext's length is related with the number of participants where the former increases at least linearly in the latter. In the scheme of Clear and McGoldrick [3], an encryption of a message contains a universal mask U generated by another public-key encryption scheme. Also, the ratio of the size of the ciphertext under multiple keys and that under single key grow quadratically with an increase in the number of the associated participants. Afterwards, Mukherjee and Wichs [2] proposed an optimized scheme with a simple generation of the universal mask. However, there is still auxiliary information in the encryption algorithm and the ratio remains quadratic. Following the previous works, there are two independent researches about multikey fully homomorphic encryption introduced by Brakerski and Perlman [5] and Peikert and Shiehian [4], respectively. In the former scheme, although the authors replaced the algorithm of the universal mask with the bootstrapping algorithm, the ciphertext's growth rate was still linear and their evaluation keys were generated by the previous multikey fully homomorphic encryption schemes. There are two versions in the paper in [4]. In the first scheme, the encryption of a message contains a commitment of the message and an encryption of the randomness used in the former commitment algorithm. The ratio becomes linear. In the second one, the encryption algorithm only outputs a ciphertext of a message, but the ratio becomes quadratic and the evaluation keys are generated by the first scheme. In [13], the growth rate is quadratic and the output of the encryption algorithm also contains auxiliary information except a ciphertext of a plaintext. Recently, Chen et al. [6] proposed a multikey FHE scheme based on the ring-LWE assumption, in which their ciphertext-extension algorithm only generates the evaluated keys for the scheme with multiple keys but the size of the ciphertext under multiple keys also has a relationship with the number of associated parties.

3. Preliminary

3.1. Learning with Errors, SIVP, and GapSVP. Regev firstly introduced the Learning with Errors (LWE) problem in 2005 and showed that the hardness of LWE can be reduced quantum to the lattice hard problems. Then, Peikert introduced an efficient classical reduction between LWE and the lattice intractable problems. The details are given below.

Definition 1. (Learning with Errors). Let λ be the security parameter, let $n = n(\lambda)$ be an integer dimension of a lattice, let $q = q(\lambda) \geq 2$ be an integer, and let $\chi = \chi(\lambda)$ be an error distribution over $\mathbb{Z}$.

(i) (Searchable LWE) Sample $\mathbf{s} \longleftarrow \mathbb{Z}_q^n$ uniformly and then draw $\mathbf{a}_i \longleftarrow \mathbb{Z}_q^n$ uniformly, $e_i \longleftarrow \chi$. Set $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i$. The searchable LWE is to find $\mathbf{s}$, given $m = m(\lambda)$ samples $\{(\mathbf{a}_i, b_i)\}_{i=1}^m$, called $\mathrm{LWE}_{n,m,q,\chi}$.

(ii) (Decision LWE) The decision LWE, denoted as $\mathrm{LWE}_{n,q,\chi}$, is to distinguish two distributions: The first one is a uniform distribution over $\mathbb{Z}_q^{n+1}$. The second is that one first samples $\mathbf{s} \longleftarrow \mathbb{Z}_q^n$ and then draws $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ by sampling $\mathbf{a}_i \longleftarrow \mathbb{Z}_q^n$ uniformly, $e_i \longleftarrow \chi$, and setting $b_i = \langle \mathbf{a}_i, \mathbf{s}_i \rangle + e_i$.

The Learning with Errors (LWE) assumption is that $\mathrm{LWE}_{n,m,q,\chi}$ ($\mathrm{LWE}_{n,q,\chi}$) is intractable.

Definition 2. ($\mathrm{SIVP}_{\gamma(n)}$). Let Λ be an n−dimension lattice. The $\mathrm{SIVP}_{\gamma(n)}$ problem is to output n linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ such that $\max_i\{v_i\} \leq \gamma(n) \cdot \lambda_n$, where $\lambda_n = \min_r\{r\colon \dim(\mathrm{span}(B(0,r) \cap \Lambda)) \geq n\}$.

Definition 3. ($\mathrm{GapSVP}_{\gamma(n)}$). Let Λ be an n−dimension lattice and let d be a real number. $\mathrm{GapSVP}_{\gamma(n)}$ is to distinguish whether $\lambda_1 < d$ or $\lambda_1 \geq \gamma(n) \cdot d$, where λ_1 is the length of the shortest vector in Λ.

Definition 4. (B-bounded distributions). A distribution ensemble $\{\chi_n\}_{n\in\mathbb{N}}$ over the integers is called B-bounded distribution if

$$\Pr_{e \longleftarrow \chi_n}[|e| > B] = \mathrm{negl}(n). \tag{1}$$

Theorem 1. *Let $q = q(n)$ be either a prime power or a product of small (size poly(n)) distinct primes, $B \geq \omega(\log n) \cdot \sqrt{n}$, and χ is an efficient sampleable B-bounded distribution. If there exists an efficient algorithm solving the $LWE_{n,q,\chi}$ problem, then*

There is an efficient quantum algorithm for $GapSVP_{\widetilde{O}(nq/B)}$ on any n-dimension lattice

There is an efficient classical algorithm that solves $GapSVP_{\widetilde{O}(nq/B)}$ on any n-dimension lattice

In both cases, if one also considers solving $LWE_{n,q,\chi}$ with subpolynomial advantage, then request $B \geq \widetilde{O}(n)$ and $\gamma(n) \geq \widetilde{O}(n^{1.5}q/B)$.

3.2. Fully Homomorphic Encryption. A fully homomorphic encryption is a tuple of algorithms (**Gen**, **Enc**, **Dec**, **Eval**) described as follows:

(pk, sk, evk) $\longleftarrow$ **Gen**(1^λ): on the security parameter λ, output a public key pk, a secret key sk, and a public evaluation key evk.

$c \longleftarrow$ **Enc**(pk, μ): encrypt a message μ from the plaintext space and output a ciphertext c.

$\mu \longleftarrow$ **De c**(sk, c): decrypt a valid ciphertext c and output a corresponding message μ; otherwise, output a special symbol $\perp$.

$c_f \longleftarrow$ **Eval**$(\mathrm{evk}, f, c_1, \ldots, c_l)$: input the public evaluation key evk, a function f, and a sequence of ciphertexts $c_1, \ldots, c_l$ which are responding to the sequence of plaintexts $\mu_1, \ldots, \mu_l$; output a valid ciphertext c_f responding to the message $f(\mu_1, \ldots, \mu_l)$.

We say that a scheme $\Pi = $ (**Gen**, **Enc**, **De c**, **Eval**) is fully homomorphic if it satisfies the following properties:

Homomorphism: denote a class of all arithmetic circuits over GF(2) as $\mathbb{C}$. If for arbitrary circuit $f \in \mathbb{C}$, the following inequation holds:

$$\Pr[\mathbf{Dec}(\mathrm{sk}, \mathbf{Eval}(\mathrm{evk}, f, c_1, \ldots, c_l)) \neq f(\mu_1, \ldots, \mu_l)] = \mathrm{negl}(\lambda). \tag{2}$$

Compactness: if there exists a polynomial $p = \mathrm{poly}(\lambda)$, it holds that the output length of **Eval** is at most $p(\lambda)$ bits without relation to the function f or the numbers of inputs.

3.3. Multikey Fully Homomorphic Encryption

Definition 5. (multikey FHE). A multikey FHE is a tuple of algorithms (**Setup**, **Keygen**, **Encrypt**, **Expand**, **Eval**, **Decrypt**) described as follows:

params $\longleftarrow$ **Setup**$(1^\lambda, 1^d)$: on the security parameter λ and the circuit depth d, the setup algorithm outputs the system parameters params. We assume that all the other algorithms take params as an input implicitly.

$(\mathrm{sk}, \mathrm{pk}) \longleftarrow$ **Keygen** (params): generate secret key sk and public key pk.

$c \longleftarrow$ **Encrypt** (pk, μ): take public key pk and a message μ as an input and output for a ciphertext c.

$\widehat{c} \longleftarrow$ **Expand** $(\mathrm{pk}_1, \ldots, \mathrm{pk}_N, i, c)$: on a sequence of N public keys and a fresh ciphertext c under the i–th key pk_i, it outputs an expanded ciphertext $\widehat{c}$.

$\widehat{c}$: =**Eval** $(\mathrm{params}, \mathscr{C}, (\widehat{c}_1, \ldots, \widehat{c}_l))$: given a Boolean circuit $\mathscr{C}$ of depth $\leq d$ along with l expanded ciphertexts $\widehat{c}_1, \ldots, \widehat{c}_l$, output an evaluated ciphertext $\widehat{c}$.

μ: =**Decrypt** $(\mathrm{params}, (\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c})$: take some ciphertext $\widehat{c}$ and a sequence of N secret keys as an input and output a message μ.

The following properties hold:

Semantic Security of Encryption. For any polynomial $d = d(\lambda)$ and any two messages μ_0, μ_1, the distribution (params, pk, **Encrypt**(pk, μ_0)) is computationally indistinguishable from the distribution (params, pk, **Encrypt** (pk, μ_1)), where params $\longleftarrow$ **Setup** $(1^\lambda, 1^d)$, $(\mathrm{sk}, \mathrm{pk}) \longleftarrow$ **Keygen** (params).

Correctness and Compactness. Let params $\longleftarrow$ **Setup** $(1^\lambda, 1^d)$. Consider any sequence of N correctly generated key pairs $\{(\mathrm{pk}_i, \mathrm{sk}_i) \longleftarrow \mathbf{Keygen}(\mathrm{params})\}_{i\in[N]}$ and l–tuple of messages $(\mu_1, \ldots, \mu_l)$. For any sequence of indices $(I_1, \ldots, I_l)$ where each $I_i \in [N]$, let $\left\{c_i \longleftarrow \mathbf{Encrypt}(\mathrm{pk}_{I_i}, \mu_i)\right\}_{i\in[l]}$ be encryptions of the messages μ_i under the I_i–th public key and let $\{\widehat{c}_i \longleftarrow$ **Expan d**$((\mathrm{pk}_1, \ldots, \mathrm{pk}_N), I_i, c_i)\}_{i\in[l]}$ be the corresponding expanded ciphertexts. Let $\mathscr{C}$ be any Boolean circuit of depth $\leq d$ and let $\widehat{c}$: $=$ **Eval**$(\mathscr{C}, (\widehat{c}_1, \ldots, \widehat{c}_l))$ be the evaluated ciphertext. Then the following holds:

Correctness of Expansion. $\forall i \in [l]$, **Decrypt** $((\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c}_i) = \mu_i$.

Correctness of Evaluation. **Decrypt** $((\mathrm{sk}_1, \ldots, \mathrm{sk}_N), \widehat{c}) = \mathscr{C}(\mu_1, \ldots, \mu_l)$.

Compactness. There exists a polynomial $p(\cdot)$ such as $|\widehat{c}| \leq p(\lambda, d, N)$. In other words, the size of $\widehat{c}$ should be independent of $\mathscr{C}$ and l but can depend on λ, d, N.

4. A Scheme of Evaluation on Two-Key Ciphertexts for Any Polynomial

In this section, we formally describe our fully homomorphic encryption scheme. At the beginning, we introduce three operations used in the encryption algorithm for slow noise growth. Consider three vectors $\mathbf{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{Z}_q^n$, $\alpha = (\alpha_0, \ldots, \alpha_{N-1}) \in \{0,1\}^N$, and $\beta = (\beta_0, \ldots, \beta_{N-1}) \in \mathbb{Z}_q^N$.

BitDecomp$(a) = (a_{0,0}, a_{0,1}, \ldots, a_{0,l-1}, a_{1,0}, \ldots, a_{1,} l-1, \ldots, a_{n-1,l-1})$, where $a_{i,j}$ is the j-th element of the binary representation of a_i.

BitDecomp$^{-1}(\alpha) = (\sum_{i=0}^{l-1} 2^i \alpha_i, \sum_{i=l}^{2l-1} 2^{i-l} \alpha_i, \ldots, \sum_{i=(n-1)l}^{N-1} 2^{i-(n-1)l} \alpha_i)$, where $\alpha \in \{0,1\}^N$.

Flatten$(\beta) = $ BitDecomp(BitDecomp$^{-1}(\beta)$).

We can see that BitDecomp$(\cdot)$ expands each element of a vector to its binary representation, BitDecomp$^{-1}(\cdot)$ can be seen as the inverse operation of BitDecomp$(\cdot)$, and it makes each l element of a vector to a number in $\mathbb{Z}_q$. These three operations on a matrix are that they are performed on each column vector of the matrix. That is, $\mathrm{BitDecomp}(A) = \begin{bmatrix} \mathrm{BitDecomp}(A_0) \\ \vdots \\ \mathrm{BitDecomp}(A_{n-1}) \end{bmatrix}$. BitDecomp$^{-1}(\cdot)$ and Flatten$(\cdot)$ on a matrix are similar to that.

Our scheme consists of the following probabilistic polynomial time algorithms (**Setup**, **Gen**, **Enc**, **Dec**, **Add**, **Mult**, **Add2**, **Mult2**, and **Dec2**).

Setup $(1^\lambda, 1^L)$: let λ be the security parameter and let L be the max circuit depth. Choose appropriate LWE parameters: modulus $q = q(\lambda, L)$, lattice dimension $n = n(\lambda, L)$, and error distribution $\chi = \chi(\lambda, L)$. Choose parameter $m = O(n \log q)$. Set params $= (q, n, \chi, m)$. Let $l = \lfloor \log q \rfloor + 1$ and $N = n \times l$.

Gen (params): choose randomly $\mathbf{t} \longleftarrow \mathbb{Z}_q^{n-1}$. Choose a random matrix $B \longleftarrow \mathbb{Z}_q^{m\times(n-1)}$ and a vector $\mathbf{e} \longleftarrow \chi^m$. Set $\mathbf{b} = B \cdot \mathbf{t} + \mathbf{e}$. Output the secret key $\mathrm{sk} = \mathbf{s} = (1, -t_1, \ldots, -t_{n-1}) \in \mathbb{Z}_q^n$ and the public key

$\mathrm{pk} = A = [\mathbf{b}|B]$. Let $\mathbf{v} = \mathrm{Powerof2}(\mathbf{s})$ (note that $A \cdot \mathbf{s} = \mathbf{e}$.)

Enc $(\mathrm{params}, \mathrm{pk}, \mu)$: choose randomly a matrix $R \longleftarrow \{0,1\}^{N\times m}$. Then encrypt the message μ as follows:

$$C = \mathrm{Flatten}\left(\mu \cdot I_N + \mathrm{BitDecomp}\left(R \cdot A\right)\right) \in \mathbb{Z}_q^{N\times N}. \tag{3}$$

Output the ciphertext C.

Dec $(\mathrm{params}, \mathrm{sk}, C)$: let $v_i \in ((q/4), (q/2)]$. Output $\mu = \lfloor \langle C_i, \mathbf{v}\rangle / v_i \rceil$.

Add $(\mathrm{params}, \mathrm{pk}, C_1, C_2)$: to add two ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N\times N}$, output $\mathrm{Flatten}(C_1 + C_2)$.

Mult $(\mathrm{params}, \mathrm{pk}, C_1, C_2)$: to multiply two ciphertexts $C_1, C_2 \in \mathbb{Z}_q^{N\times N}$, output $\mathrm{Flatten}(C_1 \cdot C_2)$.

Mult2 $(\mathrm{params}, \mathrm{pk}_1, \mathrm{pk}_2, C_1, C_2)$: these two keys are independently generated from the algorithm Gen () on the common parameters. If C_1 is not encrypted under pk_1 or C_2 is not under pk_2, then output $\perp$. Otherwise, output $C_{1,l-1} \cdot C_{2,l-1}^T$.

Add2 $(\mathrm{params}, \mathrm{pk}_1, \mathrm{pk}_2, C_1, C_2)$: if C_1 is not encrypted under pk_1 or C_2 is not under pk_2, then output $\perp$. Otherwise, set C_1' and C_2' as encryptions of message 1 under pk_1 and pk_2, respectively, and output $C_{1,l-1} \cdot (C_{2,l-1}')^T + C_{1,l-1}' \cdot C_{2,l-1}^T$.

Dec2 $(\mathrm{parmas}, C, \mathrm{sk}_1, \mathrm{sk}_2)$: if C is an evaluated ciphertext from two ciphertexts under the public keys pk_1 and pk_2, respectively, then the first secret key sk_1 holder computes $\mathrm{tempc}_1 = \mathbf{v}_1^T \cdot C$ and sends it to the sk_2 holder. Similarly, the sk_2 holder computes $\mathrm{tempc}_2 = C \cdot \mathbf{v}_2$ and sends it to the first holder. Then, the sk_1 holder outputs $\mathbf{v}_1^T \cdot \mathrm{tempc}_2$ and the sk_2 holder outputs $\mathrm{tempc}_1 \cdot \mathbf{v}_2$.

The evaluation algorithm **Eval**$(\cdot)$ that performs a depth-L circuit computations on polynomial GSW ciphertexts can be composed of **Add** and **Multi** operations.

5. Evaluation on Two-Key FHE Ciphertexts

5.1. Multiplication. Assume that $\mathbf{C}_1$ is a GSW ciphertext of the message μ_1 under the public key pk_1 and $\mathbf{C}_2$ is that of μ_2 under pk_2. $\mathbf{s}_1$ and $\mathbf{s}_2$ are secret keys corresponding to pk_1 and pk_2, respectively. Set $\mathbf{v}_i$ =Powerof2$(\mathbf{s}_i)$, $i = 1, 2$. This function Powerof2() transforms a vector $(a_0, \ldots, a_{n-1})$ into a new vector $(a_0, 2a_0, \ldots, 2^{l-1}a_0, \ldots, a_{n-1}, \ldots, 2^{l-1}a_{n-1})$, where l is the length of the binary representation of the modulus q.

Mult2$(\mathbf{C}_1, \mathbf{C}_2) = \mathbf{c}_1 \cdot \mathbf{c}_2^T$, where $\mathbf{c}_i = \mathbf{C}_i[l-1, \cdot]$, $i = 1, 2$.

De c2$(\mathbf{v}_1, \mathbf{v}_2, \mathbf{C})$: Compute $(\lfloor \mathbf{v}_1^T \cdot \mathbf{C}/2^{l-1} \rceil)^T + c'$, denoted as **tc**, where c' is the $(l-1)$–th row of a ciphertext of a message 0 under the secret key $\mathbf{v}_2$ such that $\lfloor (\langle c', \mathbf{v}_2\rangle / 2^{l-1}) \rceil$. Output $\lfloor \langle \mathbf{tc}, \mathbf{v}_2 \rangle / 2^{l-1} \rceil$.

Theorem 2. *Suppose that* $\mathbf{C}_1, \mathbf{C}_2$ *are ciphertexts under the secret keys* $\mathbf{v}_1, \mathbf{v}_2$, *respectively. If* $\mathbf{C}$ *is obtained from* **Mult**2$(\mathbf{C}_1, \mathbf{C}_2)$ *or* **A dd** 2$(\mathbf{C}_1, \mathbf{C}_2)$, *the probability of the decryption algorithm* **De c**2$(\cdot)$ *on inputs* $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{C}\}$ *running correctly is negligible. That is, there exists a negligible function negl*$(\cdot)$ *on the security parameter* λ, *satisfying the following inequation:*

$$\Pr\left[\mathbf{Dec}2(\mathbf{v}_1, \mathbf{v}_2, \mathbf{C}) \neq \mu_1\mu_2 \;\middle|\; \begin{array}{c} (\mathrm{pk}_i, \mathrm{sk}_i) \longleftarrow \mathbf{GSW}.\mathbf{Gen}\left(1^\lambda\right), \quad i = 1, 2; \\ v_i = \mathrm{Powerof}\,2(\mathrm{sk}_i), \quad i = 1, 2; \\ \mu_1, \mu_2 \longleftarrow \{0,1\}, \mathbf{C}_i = \mathbf{GSW}.\mathbf{Enc}(\mathrm{pk}_i, \mu_i); \\ \mathbf{C} = \mathbf{Mult}2(\mathbf{C}_1, \mathbf{C}_2), \end{array}\right] \leq \mathrm{negl}(\lambda). \tag{4}$$

Proof. Obviously, $\mathbf{C}_i \cdot \mathbf{v}_i = \mu_i \cdot \mathbf{v}_i + \mathbf{e}_i$, $i = 1, 2$. We also know that the first l elements of $\mathbf{v}_i$ are $(1, 2, \ldots, 2^{l-1})$. Thus, we can decrypt the ciphertext C_i as $\mu_i = \lfloor \langle C_i[l-1, \cdot], \mathbf{v}_i \rangle / 2^{l-1} \rceil$. Set $\mathbf{c}_i = C_i[l-1, \cdot]$ and $e_i = \mathbf{e}_i[l-1]$, $i = 1, 2$. So, $\langle \mathbf{c}_i, \mathbf{v}_i \rangle = \mu_i \cdot 2^{l-1} + e_i$. Running the first part of the decryption algorithm, we can obtain that $\mathbf{tc} = (\lfloor \mathbf{v}_1^T \cdot \mathbf{C}/2^{l-1} \rceil)^T + c' = (\lfloor \mathbf{v}_1^T \cdot \mathbf{c}_1 \cdot \mathbf{c}_2^T / 2^{l-1} \rceil)^T + c' = (\lfloor \mathbf{c}_2^T \cdot (\mu_1 \cdot 2^{l-1} + e_i)/2^{l-1} \rceil)^T + c' = \mu_1 \mathbf{c}_2 + c'$. After the second part, we can get $\lfloor \langle \mathbf{tc}, \mathbf{v}_2 \rangle / 2^{l-1} \rceil = \lfloor \mu_1 \langle \mathbf{c}_2, \mathbf{v}_2 \rangle / 2^{l-1} + \langle c', \mathbf{v}_2 \rangle / 2^{l-1} \rceil = \mu_1\mu_2$ That is to say, one-time multiplication on two ciphertexts under different secret keys only increases doubly the size of noise because the noise in the intermediate ciphertext **tc** can be viewed as that in an addition to two GSW ciphertexts under the same secret key. Therefore, the ciphertexts obtained from this multiplication algorithm can be decrypted correctly.

We can easily find that one-time multiplication causes a double increase of noise. Thus, scaling up the parameters or appending something auxiliary is undesired. We can directly perform one-time multiplication on two ciphertexts encrypted by two different keys without adjusting anything of the original GSW scheme.

5.2. Addition. We can achieve the *Addition* operation by using the operation *Multiplication*. That is, **A dd** 2$(\mathbf{C}_1, \mathbf{C}_2) = \mathbf{Mult}2(\mathbf{C}_1, \overline{\mathbf{C}}_2) + \mathbf{Mult}2(\overline{\mathbf{C}}_1, \mathbf{C}_2)$, where $\overline{\mathbf{C}}_i$ is a ciphertext of message 1 under the secret key $\mathbf{v}_i$, $i = 1, 2$.

According to Theorem 2, after one-time operation *Multiplication* on two ciphertexts under different secret keys, the noise increases doubly. Thus, one-time operation

Addition causes the noise to increase quadruply, which is faster than that of *Multiplication*. It is not hard to find that the ciphertext $\overline{\mathbf{C}}_i$ is unnecessary to preserve the privacy of the plaintext, an exact number 1. Therefore, when constructing $\overline{\mathbf{C}}_i$, we can set the randomness to zero. That is to say, $\overline{\mathbf{C}}_i$ is a special "ciphertext" of the plaintext 1 without noise. This change makes both the operations *Addition* and *Multiplication* have the same growth of the noise.

Note that the **Add2** operation not only supports the input of two ciphertexts under different secret keys but also processes the input of one ciphertext obtained from the **Add2** or **Mult2** procedure and one ciphertext under a single key as well as the input of two former-type ciphertexts. The following are the details of the operation.

Assume that C' is output by the **Add2** or **Mult2** procedure and $\mathbf{C}$ is a ciphertext under the secret key $\mathbf{v}_{b+1}$, where $b \longleftarrow \{0,1\}$. Then $\mathbf{A\ DD}(\mathbf{C}, C') = \mathbf{Mult2}(\mathbf{C}, \overline{\mathbf{C}}) + C'$, where $\overline{\mathbf{C}}$ is a ciphertext of message 1 under the secret key $\mathbf{v}_{\bar{b}+1}$.

Assume that $\mathbf{C}, C'$ are both output by the **Add2** or **Mult2** procedure. Then, $\mathbf{A\ DD}(\mathbf{C}, C') = \mathbf{C} + C'$. It also can extend to the case of the input of polynomial ciphertexts from the **Add2** or **Mult2** procedure.

5.3. Evaluation of Any Polynomial Function. Assume that f is an arbitrary polynomial function of $u + v$ inputs, denoted as $x_1, \ldots, x_u, y_1, \ldots, y_v$ and can be rewritten as $f(x_1, \ldots, y_v) = \sum_{i=1}^{w} g_i(x_1, \ldots, x_u) f_i(y_1, \ldots, y_v)$, where g_i and f_i are all L−bounded-depth circuits. Now, we have $u + v$ ciphertexts denoted as $\mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}$ under the public key pk_1 and $\mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}$ under the public key pk_2. So,

$$\begin{aligned}
&\mathbf{Eval}\Big(\mathrm{pk}_1, \mathrm{pk}_2, \mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}, \mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}, \mathbb{C}_f\Big) \\
&= \sum_{i=1}^{w} \mathbf{Mult2}\Big(\mathbf{GSW.Eval}\Big(\mathrm{pk}_1, \mathbf{C}_{1,1}, \ldots, \mathbf{C}_{1,u}, \mathbb{C}_{g_i}\Big), \mathbf{GSW.Eval}\Big(\mathrm{pk}_2, \mathbf{C}_{2,1}, \ldots, \mathbf{C}_{2,v}, \mathbb{C}_{f_i}\Big)\Big) \\
&= \mathbf{ADD}\Big(\mathbf{Mult2}\Big(C_{g_1}, C_{f_1}\Big), \ldots, \mathbf{Mult2}\Big(C_{g_w}, C_{f_w}\Big)\Big),
\end{aligned} \tag{5}$$

where $C_{g_i} = \mathrm{Eval}(\mathrm{pk}_1, C_{1,1}, \ldots, C_{1,u}, g_i)$ and $C_{f_i} = \mathrm{Eval}(\mathrm{pk}_2, C_{2,1}, \ldots, C_{2,v}, f_i)$.

Because g_i and f_i are all L-bounded-depth circuits, $\mathbf{C}_{g_i}$ and $\mathbf{C}_{f_i}$ can be decrypted correctly by the secret keys sk_1 and sk_2, respectively. The operations *Addition* and *Multiplication* both cause the noise to increase linearly. Therefore, the output of the algorithm **Eval** can be decrypted correctly.

6. Analysis

6.1. Correctness. Suppose that $\mathbf{C}_1$ and $\mathbf{C}_2$ are GSW ciphertexts of the plaintexts μ_1 and μ_2 under the public keys pk_1 and pk_2, respectively, so that $\mathbf{C}_i \cdot \mathbf{v}_i = \mu_i \cdot \mathbf{v}_i + \mathbf{small}_i$. These two ciphertexts are possibly fresh GSW ciphertexts and also can be evaluated ciphertexts through a circuit of the depth less than L. Also, a fresh GSW ciphertext has a B−bounded noise, namely, $|\mathbf{small}|_\infty \le B$. The error is bounded by $B(N + 1)$ after one homomorphic operation. So, $\mathbf{C}_i$ is a ciphertext with $B(N + 1)^L$−bounded noise. From the simple analysis in the front section, the noise in **Mult2** (C_1, C_2) is bounded by $2B(N + 1)^L$. Moreover, the noise in the addition of C_1 and C_2 increases linearly as the same as that of the *Multiplication*. So, finishing one-time homomorphic operation on two ciphertexts under different encryption keys, the noise grows up to $2B(N + 1)^L$. We only discuss one multiplication operation on two ciphertexts under different keys and polynomial additions on two multiplied ciphertexts. Thus, we assume that there are polynomial additions $w = \mathrm{poly}(\lambda, L)$. The final evaluated ciphertext is bounded $2wB(N + 1)^L$. As long as this bound is less than $q/8$, we can decrypt the evaluated ciphertext correctly. We just set $B(N + 1)^L \le (1/4)\sqrt{q/w}$. Then, it satisfies $B(N + 1)^L \le q/8$ so that GSW ciphertexts can be decrypted correctly. Also, $2Bw(N + 1)^L \le q/8$. We can decrypt correctly evaluated ciphertexts through quadratic computations on ciphertexts under two different keys. Now, we conclude this in the following theorem.

Theorem 3. *Given the parameters, a modulus q, a lattice dimension n, a B−bounded distribution χ, and the max circuit-depth L, set $N = n \times (\lfloor \log q \rfloor + t1)$. If $B(N + 1)^L \le q/8$, we can decrypt correctly a ciphertext from evaluating a depth-L circuit.*

Theorem 4. *Given the above parameters q, n, χ, B, L, N, and w, that is, the number of additions of a quadratic function, if $B(N + 1)^L \le (1/4)\sqrt{q/w}$, we can decrypt a ciphertext, that is, from performing a quadratic computations on fresh GSW ciphertexts under two different keys or evaluated ciphertexts through a depth-L circuit under two different keys.*

6.2. Security. The security of our scheme is dependent on that of the GSW scheme. The inputs of the evaluation algorithm are just the GSW-type ciphertexts, two public keys, and some common parameters without other information of private inputs. Thus, this process reveals no knowledge. In the process of the decryption, the output of the first part is indistinguishable with the uniform distribution because it adds a fresh ciphertext of message 0 and introduces a new noise in the intermediate result. So, we can conclude the following theorem.

Theorem 5. *Assume that the GSW scheme is semantically secure, and so does our scheme. That is, if there exists a* *probabilistic polynomial time adversary $\mathscr{A}$ which can distinguish the distribution of the ciphertext of the GSW scheme and the uniform distribution, we can construct another probabilistic polynomial time adversary $\mathscr{B}$ which can distinguish the distribution of the ciphertext of our scheme and the uniform distribution.*

7. Conclusion

In this paper, we present an efficient algorithm of secure computation on ciphertexts under two different keys. In previous works, when evaluating multikey ciphertexts, the size of the ciphertext grows with the number of participants at a more or less linear rate. Although the size of the ciphertext remains invariant, it also provides auxiliary information of the plaintexts. We wanted to evaluate directly on the GSW ciphertexts from two parties without any auxiliary information or interaction between them. We designed a scheme in which one can directly perform any polynomial function on the GSW ciphertexts under two different keys.

Data Availability

No data were used to support this study.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC*, pp. 1219–1234, New York, NY, USA, May 2012.

[2] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key FHE," in *Proceedings of the Part II 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 735–763, Vienna, Austria, May 2016.

[3] M. Clear and C. McGoldrick, "Multi-identity and multi-key leveled FHE from learning with errors," in *Proceedings of the Part II 35th Annual Cryptology Conference*, pp. 630–656, Santa Barbara, CA, USA, August 2015.

[4] C. Peikert and S. Shiehian, "Multi-key FHE from LWE, revisited," in *Proceedings of the Part II Theory of Cryptography-14th International Conference, TCC 2016-B*, pp. 217–238, Beijing, China, October 2016.

[5] Z. Brakerski and R. Perlman, "Lattice-based fully dynamic multi-key FHE with short ciphertexts," in *Proceedings of the Part I Advances in Cryptology-CRYPTO 2016-36th Annual International Cryptology Conference*, pp. 190–213, Santa Barbara, CA, USA, August 2016.

[6] L. Chen, Z. Zhang, and X. Wang, "Batched multi-hop multi-key FHE from ring-LWE with compact ciphertext extension," in *Proceedings of the Part II Theory of Cryptography-15th International Conference, TCC*, pp. 597–627, Baltimore, MD, USA, November 2017.

[7] W. Chongchitmate and R. Ostrovsky, "Circuit-private multi-key FHE," in *Proceedings of the Part II Public-Key Cryptography-PKC 2017-20th IACR International Conference on Practice and Theory in Public-Key Cryptography*, pp. 241–270, Amsterdam, The Netherlands, March 2017.

[8] Z. Li, C. Ma, and H. Zhou, "Multi-key FHE for multi-bit messages," *Science China Information Sciences*, vol. 61, no. 2, 2018.

[9] H. Chen, I. Chillotti, and Y. Song, "Multi-key homomophic encryption from TFHE," *IACR Cryptology ePrint Archive*, vol. 116, 2019.

[10] T. Zhou, N. Li, X. Yang, Y. Han, and W. Liu, "Efficient multi-key FHE with short extended ciphertexts and less public parameters," *IACR Cryptology ePrint Archive*, vol. 1054, 2018.

[11] B. Jiang and Y. Zhang, "Privacy-preserving min and k-th min computations with fully homomorphic encryption," in *Proceedings of the 34th IEEE International Performance Computing and Communications Conference, IPCCC*, pp. 1–8, IEEE Computer Society, Nanjing, China, December 2015.

[12] B. Jiang and Y. Zhang, "Securely min and *k*-th min computations with fully homomorphic encryption," *Science China Information Sciences*, vol. 61, no. 5, 2018.

[13] Z. Brakerski, S. Halevi, and A. Polychroniadou, "Four round secure computation without setup," in *Proceedings of the Part I Theory of Cryptography-15th International Conference*, pp. 645–677, Baltimore, MD, USA, November 2017.

[14] G. Craig, S. Amit, and B. Waters, "Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based," in *Proceedings of the Part I Advances in Cryptology-CRYPTO 2013-33rd Annual Cryptology Conference*, Santa Barbara, CA, USA, August 2013.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6696149, 8 pages
https://doi.org/10.1155/2021/6696149

Research Article

A Privacy-Preserving Caching Scheme for Device-to-Device Communications

Yuqing Zhong,[1] **Zhaohua Li**,[2] **and Liping Liao**[3]

[1]*Communication Research Center, Guangzhou Power Supply Bureau, Guangzhou 510600, China*
[2]*Guangdong Electric Power Design and Research Institute, China Energy Construction Group, Guangzhou 510600, China*
[3]*Guangdong Polytechnic Normal University, Guangzhou, China*

Correspondence should be addressed to Liping Liao; liping1110@hotmail.com

Received 26 October 2020; Revised 25 November 2020; Accepted 19 January 2021; Published 8 February 2021

Academic Editor: Weizhi Meng

Copyright © 2021 Yuqing Zhong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With device-to-device (D2D) communication, user equipment can share data with each other without the involvement of network infrastructures. In order to maintain the Quality of Service (QoS) and Quality of Experience (QoE) for user applications in D2D communications, most existing schemes use proactive content caching that needs to predict content popularity before making caching decisions which may result in privacy leakage, since the information of users is collected to train a deep learning-based model to predict content popularity. Therefore, it is crucial to guarantee secure data collection in machine learning-based framework. In this paper, we propose a privacy-preserving D2D caching scheme with a passive content caching strategy based on node importance, which can deliver more efficient caching and prevent the potential leakage of user privacy. The scheme is based on softwaredefined networking (SDN), in which the controller is responsible for calculating node importance of devices according to the information of requests and encounters collected by SDN switches. Base station will decide which device can establish reliable and secure communication with content requester based on historical information. The simulation results show that the proposed strategy can outperform other D2D caching strategies in terms of cache hit rate and data rate.

1. Introduction

During the recent years, with the rapid development of the mobile Internet, mobile data traffic has increased exponentially. A Cisco VNI report predicted that 79% of global mobile data traffic will be mainly generated from access to video content by 2022 [1]. The current wireless network is facing huge challenges. To reduce the backhaul traffic and base station load, device-to-device (D2D) communication technology has emerged. User devices can directly establish D2D communication links with other devices within the communication range in the D2D communication network without the use of base stations or other access points.

Research results have shown that most mobile traffic is generated from repeated access to popular content [2, 3]. By deploying caches in the core network [4], access network [5–8], and user device [9, 10] in the 5G mobile communication network architecture, the popular content can be cached at the network edges, which can effectively reduce network congestion and improve network performance. The content is cached to the user devices, and a D2D caching network is constructed. Thus, user devices can share content via the D2D communication link without the use of access points, which can effectively reduce backhaul traffic and base station load. The caching mechanism can determine the caching position and caching contents, is the kernel of the D2D caching network, and determines the caching performance. However, due to the heterogeneity of D2D caching network, it is more vulnerable to security and data privacy threats. It is important to find the trade-off between the performance of security and the cost of protection. Most existing D2D caching mechanisms belong to a proactive caching mechanism that requires content popularity in advance. Before a user sends a request, the content with higher popularity is cached into the D2D caching network in advance. Some owners of mobile devices may be curious about the content cached in their devices and scan the caching content, which may result in privacy leaking [11].

During recent years, short videos distributed by users have attracted massive traffic. The QuestMobile report indicates that the time spent by users in short video applications was 5.5% of the total time spent on mobile applications in 2017. With the quick development of short video applications, such as Bilibili and TikTok, a user can distribute short videos whenever and wherever possible. Short videos can be distributed and requested in a random and burst manner. After popular videos are distributed, they are accessed by massive users in a short period. For example, one user distributed a short video on TikTok, and it was accessed 30,000 times in a couple of minutes. In such cases, a proactive caching strategy cannot predict the popularity of popular video content in time. Thus, the cached contents cannot be updated in real time, which results in massive ineffective caching and wastes network resources.

Motivated by this, we propose a privacy-preserving D2D caching scheme with passive content caching based on node importance to update the cached contents in real time, increase the cache hit rate, and preserve the user privacy. Since the proposed caching scheme adopts a passive caching strategy, the device can only cache the content that it requests, which prevents the leakage of user privacy. By using the network coding technology [12], the diversity of the cached contents can also be improved in limited cache size. It has been proved that using networking coding in content caching can improve the security and performance of the caching system [13]. The SDN switch collects the history request of the terminal devices and meeting information among devices, and the SDN controller can compute the node importance of the terminal devices using the history information. Base station decides which content holder can establish reliable and secure D2D communication link with the content requester. The cached contents can be updated in real time based on user requests and node importance.

The contributions of this paper include the following:

(i) Firstly, we introduce a privacy-preserving D2D caching scheme with passive content caching based on node importance to improve the security and performance of D2D caching network. The device with higher node importance and social trust will be selected to establish reliable and secure D2D communication link with the content requester.

(ii) Then, we define the node importance as the weighted sum of the physical intimacy and request similarity between devices, which also reflects the social trust of the device.

(iii) Finally, we evaluate the performance of the proposed D2D caching strategy and other two well-known caching strategies. The simulation results show that the caching strategy based on node importance proposed by this paper could effectively improve network performance compared to the other two caching strategies.

The remainder of the paper is organized as follows. We introduce related works in Section 2. In Section 3, we define the node importance and propose a privacy-preserving D2D caching strategy based on node importance. Simulation results are presented in Section 4. Finally, we conclude the paper in Section 5.

2. Related Work

If the popular content is cached in the user devices, it can effectively reduce the backhaul traffic and the downloading delay for users. Thus, service quality and user experience can be improved. Currently, most D2D caching mechanisms are based on a proactive caching strategy, and the content popularities are assumed to be known. Golrezaei et al. [14, 15] divided the D2D network into multiple D2D clusters, and only the devices in one cluster can establish the D2D communication link. Based on this, the authors proposed two in-cluster D2D caching strategies to improve network performance of cellular networks, including deterministic cache and random cache based on Zipf. In the deterministic caching mechanism [14], k devices can cache nonrepeated k contents with the top popularities in one virtual cluster and each device only caches one content. In the random caching mechanism based on the Zipf distribution [14], within one virtual cluster, each device can independently and randomly cache content and the popularities of the content cached in the cluster obeying Zipf distribution. Wang et al. proposed a novel D2D caching strategy based on mobile perception, which takes the mobility of users into account. The low-speed and high-speed moving user devices cache content with top popularities, and user devices with middle-speed moving cache content with lower popularities. Thus, the offloading rate can be improved [16]. Chen et al. modelled the offloading benefits and energy consumption of content holders and proposed a proactive caching strategy and user-oriented protocol to obtain higher offloading benefits with lower energy consumption [17]. Malak et al. extended the caching mechanism based on the geographical position and proposed a space-based caching strategy to improve the cache hit rate. To reduce cache redundancy and improve the diversity of cached contents, all the devices in the mutual exclusion area cannot cache the same content [18]. Wu et al. proposed a distributed D2D caching strategy that considers the characteristics of different requests and demands of physical links [19]. Besides a proactive caching strategy that assumes that the content popularity is known, partial D2D caching mechanisms predict content popularity by algorithms such as machine learning. After the future content popularities are predicted, the caching mechanism is determined. Jiang et al. modelled the D2D caching as the multiagent and multiarm gaming machine and determined the cache by using reinforcement learning to reduce downloading delay for users [20]. Li et al. optimized content caching and content distribution jointly to reduce transmission delay and power consumption. They deployed two potential recurrent neural network models, echo state network (ESN) and long short-term memory (LSTM), to predict mobility of users and future popularity of content and then determined the cached contents and cache position. The authors also proposed a content distribution mechanism based on deep reinforcement learning to improve user experience [21]. In another study [22], the authors proposed a proactive caching strategy based on the association between users and content. They predicted the content popularity by using machine learning and collaborative filtering technology. The content with higher popularities is precached to the base stations and user devices in the low peak period to alleviate the backhaul congestion. Chen and Yang proposed a D2D caching strategy based on user preference to improve the offloading rate, which predicts user preferences by using a collaborative filtering algorithm based on the model and then makes caching decisions [23].

In summary, most D2D caching strategies are based on proactive caching mechanisms and the content popularities must be known or be predicted. Then, the caching decision problem is transformed into optimization problems to find a solution. The prediction accuracy of future content popularities decides the caching performance. However, the rise of short video applications makes the proactive caching strategy fail to precisely predict content popularities and update cached content in real time. Thus, caching performance is reduced. Motivated by this, we propose a passive D2D caching strategy based on node importance, and the cached content can be updated in real time according to user requests. Moreover, network coding technology is employed, which can improve the diversity of the cached content and increase the cache hit rate and data rate without increasing the cache size.

3. System Design

3.1. Defining Node Importance. In this paper, we propose a software-defined passive D2D caching strategy (NIC) based on node importance. Different devices have different node importance in the D2D network. In this system, the SDN switch collects meeting information among user devices and information of content requested by user devices to compute the physical intimacy and request similarity among devices. The node importance is defined as the weighted sum of the physical intimacy [24] and request similarity. For the physical network layer, the user devices with higher node importance will have a higher probability to establish stable D2D communication links with other devices in the future. For content requesting, the user devices with higher node importance have a higher probability to request the same content as other devices. In other words, the user devices with higher node importance have a higher probability of providing other user devices with requested content in the future. Thus, the devices with higher importance will cache content with higher popularities, and the original blocks are cached to reduce the network coding and decoding time and computing consumption. The devices with lower importance will cache contents with lower popularities, and the coding blocks are cached to distribute contents in the network more reasonably, improve caching diversity, and increase the caching efficiency without increasing cache size. The physical intimacy between user devices indicates the probability of establishing reliable D2D communication links between two user devices in the future. When device D_i and device D_j are within the D2D communication range, the two devices may have D2D communication potential, which is recorded as one meeting between them. The duration is recorded as the meeting duration. Zhang et al. [24] proved that the user meeting time obeys the gamma distribution $\Gamma(k, \theta)$, namely, $X \sim \Gamma(k = (M_{ij}^2/I_{ij}), \theta = (I_{ij}/M_{ij}))$, wherein

$$
\begin{aligned}
M_{ij} &= \frac{f_n X_n}{N_{ij}},\\
I_{ij} &= \frac{\sum_n \left(X_n - M_{ij}\right)^2}{N_{ij}},
\end{aligned}
\tag{1}
$$

where X_n indicates the n^{th} meeting duration of device D_i and device D_j and N_{ij} indicates the meeting count between device D_i and device D_j. The physical intimacy $c_{ij} \in [0, 1]$ can be expressed as [24]

$$
\begin{aligned}
c_{ij} &= 1 - \int_0^{X_{\min}} f(u; k, \theta)\mathrm{d}u = 1 - \frac{\gamma(k, (X_{\min}/\theta))}{\Gamma(k)},\\
\gamma\left(k, \frac{X_{\min}}{\theta}\right) &= \int_0^{(X_{\min}/\theta)} t^{k-1} e^{-t}\mathrm{d}t,
\end{aligned}
\tag{2}
$$

where $X_{\min}$ is the minimal meeting duration required by two user devices to successfully transfer one file via the D2D communication link. Thus, the average physical intimacy $\overline{c_i}$ between device D_i and other devices in the D2D caching network is given by following:

$$
\overline{c_i} = \frac{\sum_{j=1}^{n} c_{ij}}{n}, \tag{3}
$$

where n is the quantity of the devices within the D2D communication range of device D_i. The higher the physical intimacy of the user device is, the higher its probability of establishing a reliable D2D communication link with other user devices in the future will be. As shown in Figure 1, the average physical intimacy $\overline{c_1}$ of user device 1 is

$$
\overline{c_1} = \frac{\sum_{j=2}^{5} c_{1j}}{4}. \tag{4}
$$

Based on the history requests of the device, the SDN controller can compute the request similarity $s_{ij} \in [0, 1]$ between user devices by using cosine similarity, which is expressed as follows:

$$
s_{ij} = \cos\left(\mathbf{w}_i, \mathbf{w}_j\right) = \frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\left\|\mathbf{w}_i\right\|_2 \left\|\mathbf{w}_j\right\|_2}, \tag{5}
$$

where $\mathbf{w}_i$ and $\mathbf{w}_j$ indicate the interest vector of device D_i and device D_j, respectively. The average request similarity $\overline{s_i}$ of device D_i and the other devices in the D2D caching network is

$$\overline{s_i} = \frac{\sum_{j=1}^{n} s_{ij}}{n}, \tag{6}$$

where n is the quantity of the devices within the D2D communication range of device D_i. The higher the average request similarity of the user device is, the higher the overlapping degree of requested contents with other devices will be. In this paper, we propose a passive caching mechanism, namely, the user device only caches the ever-requested content to satisfy other users' requests by providing other users with the desired content. The cached content in the user device with higher average request similarity will have a higher probability to be requested by other users. In this paper, we define the node importance as the weighted sum of the physical intimacy and request similarity. The devices with higher node importance will have a higher probability to provide other user devices with requested content and receive higher caching benefits. The node importance I_i of user device D_i is given by the following equation:

$$I_i = \alpha\overline{c_i} + \beta\overline{s_i}, \tag{7}$$

where $\overline{c_i}$ is the normalized average physical intimacy of device D_i, $\overline{s_i}$ is the normalized average request similarity of device D_i, and $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ are the design parameters and indicate the importance of physical intimacy and request similarity. In this paper, α and β are set as 0.5.

3.2. D2D Caching Strategy Based on Node Importance. In this system, to improve the caching efficiency, the base station divides the content into m content blocks with the same size. When device D_i requests content f, it will first send the request packet to the base station. If the D2D caching network includes this content, the base station will locate a group of user devices with hit caches and determine the caching plan according to the node importance. If I_i of device D_i is higher, then device D_i caches the original blocks. If the node importance I_i of device D_i is lower, then device D_i caches the coding blocks. This is because the device with higher node importance has higher request similarity with other devices, namely, the cached content has higher probabilities to be requested by other devices. The devices with higher node importance have a higher probability to successfully establish D2D communication links with other users and can ensure successful content transfer. Therefore, the devices with higher node importance will cache content with higher popularities and cache original content blocks to improve the cache hit rate and reduce decoding/coding time and computing consumption. The devices with lower node importance will cache contents with lower popularities and cache coding blocks. Each coding block should include all the original block information. This can improve the content diversity of the caching system without increasing the cache size.

In the D2D caching network, the base station maintains the D2D caching network information and locates a group of user device with hit caches, namely, the content holder. $\mathbf{Info}_k$ indicates the caching information of cached content f_k in the D2D caching network, and $\mathbf{Info}_k = \text{Info}_k^1, \ldots, \text{Info}_k^N$, where N is the number of user devices with cached content f_k. The content holders will be ranked by the node importance in descending order. Info_k^1 indicates the caching information of content f_k cached in the user devices with highest node importance, and the caching information is expressed as follows:

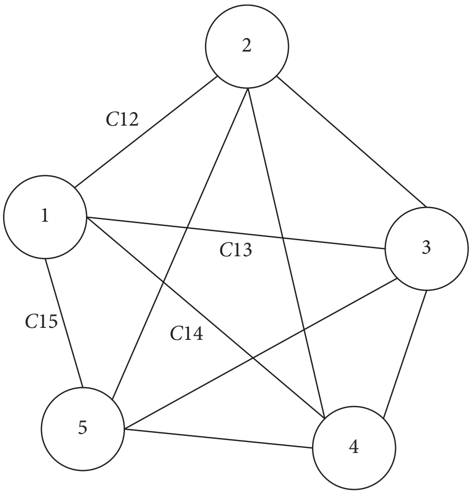


Figure 1: Example of physical intimacy.

$$\text{Info}_k^1 = \{D_i, I_i, \mathbf{V}_i, n_i\}, \tag{8}$$

where D_i is the device ID, I_i is the node importance of device D_i, n_i indicates the quantity of the cached original blocks or coding blocks, $\mathbf{V}_i = \{v_{i1}, \ldots, v_{im}\}$, and $v_{ij} \in \{0, 1\}$ indicates if the cached content block in device D_i includes the information of the original content block j. If it is included, then $v_{ij} = 1$; otherwise, $v_{ij} = 0$. When $\mathbf{V}_i$ is an all-1 vector and $n_i \neq m$, it indicates that the user device caches the coding blocks of content f_k.

When device D_i requests content f_k, the base station will find the D2D cache information table and check if the D2D caching network can satisfy the user requests. If the quantity of the content blocks cached in the D2D caching network is more than or equal to m, then the base station selects a group of content holders with higher node importance, and the selected content holders will establish D2D communication links with the device D_i to transfer corresponding content blocks. If the D2D caching network cannot satisfy the user requests, the base station will send $(m - m')$ content blocks to respond to device D_i, where m' is the quantity of the content blocks cached in the D2D caching network, as described in Algorithm 1. The complexity of Algorithm 1 is $O(n)$, where n is the number of content holders.

In this system, the base station selects a group of content holders with higher node importance and social trust to establish reliable and secure D2D communication links to transmit desired content. Moreover, the base station is also responsible for making caching decisions and instructing how the user devices cache the received content blocks. If the request is from users with higher node importance, then it indicates that caching the content in content requester will

```
Input: $\alpha, \beta$;
Output: $fq$, $nc$ //$fq$ is the cache identifier, $nc$ is the code identifier, 0 indicates the original block, and 1 is the coding block;
Initialization: $fq = 0$, $nc = 0$;
The SDN switch collects the request records and interactive information of the devices and periodically sends it to the SDN controller;
The SDN controller computes the node importance of the device according to the history information collected by SDN switches, namely, $I_i$:
While BS receives the request from device $D_i$ for content $f_k$ do
  if the node importance of device $D_i$ is higher then
    Make $fq = 1, nc = 0$;
  else
    Make $fq = 1, nc = 1$;
  end if
  if $m'$ $(m' \geq m)$ content blocks are cached in the D2D caching network then
    BS locates a group of cached content holder $D_j$ with the top node importance to respond to the user request;
    for each cached content holder $D_j$ do
      if the cached content is the original block then
        make $fq = 1$;
      else
        make $fq = 0$;
      end if
      BS sends data packets $(f_k, n_j, fq)$ to the content holder;// $n_j$ is the number of content blocks to be sent by content holder $D_j$ to the requester
      After content holder $D_j$ receives the data packets from BS, it will establish the D2D communication link with device $D_i$ and transfer the data packet $(f_k, \text{block}(s), fq)$;// block$(s)$ is the coding block or content block;
    end for
  else
    if $nc = 0$ then
      BS sends $(m - m')$ original blocks to the content requester device $D_i$, namely, $(f_k, \text{blocks}, fq)$;
    else
      BS sends $(m - m')$ coding blocks to the content requester device $D_i$, namely, $(f_k, \text{blocks}, fq)$;
    end if
  end if
end while
```

Algorithm 1: D2D caching strategy based on node importance.

bring benefits with a higher probability, i.e., higher cache hit rate. When the base station responds to the device request, it will send the original block and instruct the user devices to cache the original block. Otherwise, when the base station responds to the user request, it sends the coding blocks generated with all the original blocks and instructs the user devices to cache the coding blocks. For details, refer to Algorithm 1. To prevent users from receiving the linearly dependent coding blocks, the user device only caches the coding blocks sent by the base station and does not cache coding blocks obtained by D2D communication links.

When content requester D_i receives the content blocks from the base stations or other devices, it will decide whether to cache the received contents based on cache identifier fq. If $fq = 1$, then the content will be cached locally; otherwise, the content will not be cached. Since the cache size of the device is limited, when the caches are replaced, the user device codes the replaced content blocks into a coding block. In this way, all the original block information of this content will be reserved while the cache size is released to improve the diversity of the caching contents.

4. Experimental Results and Analysis

In the simulation test, the radius of the base station was 500 m and a total of 100 devices were provided. The maximum D2D communication distance was 100 m. The D2D communication belongs to the in-band communication, namely, the D2D communication shares the bandwidth with cellular communication [25]. The cache size of the device included $\{1, 2, 5, 8, 10, 15\}$ files, and the number of files was 500. The user request obeyed the Poisson distribution, the popularities of the content obeyed the Zipf distribution, and the Zipf parameter was $\alpha \in \{0.56, 0.8, 1, 1.2, 1.5\}$. In the simulation test, the passive caching strategy NIC in this paper was compared with the passive caching strategy, Leave Copy Everywhere (LCE), and the proactive caching strategy, Most Popular Cache (MPC). For the LCE, all the received content will be cached by requester. For the MPC, the most popular content will be cached into devices in advance. The strategies were assessed by the cache hit rate and data rate. The cache hit rate was defined as the ratio of the number of requests responded by user devices to the total requests sent by user devices, which is an important parameter to assess the performance of the caching system. When device D_i requests contents from the base station via cellular communication, the data rate $R_{B,i}$ is defined as follows [24]:

$$R_{B,i} = W\log_2\left(1 + \frac{P_B\left|h_{Bi}\right|^2}{\sum_{j'}\beta_{j'i}P_{j'}\left|h_{j'i}\right|^2 + N_0}\right). \tag{9}$$

When device D_i requests contents from device D_j via D2D communication, the data rate is defined as follows [24]:

$$R_{j,i} = W\log_2\left(1 + \frac{P_j\left|h_{ji}\right|^2}{P_B\left|h_{Bi}\right|^2 + \sum_{j'\neq j}\beta_{j'i}P_{j'}\left|h_{j'i}\right|^2 + N_0}\right), \tag{10}$$

where P_B, P_j, and $P_{j'}$ indicate the transmission power of the base station, device D_j, and device $D_{j'}$; N_0 is the Gauss white noise; $|h_{Bi}|^2$ and $|h_{j'i}|^2$ are the path loss and are related to communication distance; $\beta_{j'i} = 1$ indicates interference; and $\beta_{j'i} = 0$ indicates no interference.

The influences of the cache size on the three caching mechanisms are shown in Figures 2 and 3. As shown in Figure 2, the cache hit rate of the three caching mechanisms increased with the growth of the cache size. This is because with the growth of the cache size of the user device, more files can be cached in the D2D caching network to make more requests responded to by other terminals. In this case, it can reduce the base station load and backhaul traffic to improve the data rate, as shown in Figure 3. As shown in Figures 2 and 3, the cache hit rate and data rate of the NIC caching strategy proposed in this paper were higher than those of the other two caching strategies. The NIC strategy deploys caches and implements differential caching strategies according to the content popularities and node importance. Thus, the content can be distributed more reasonably in the D2D caching network and more content can be obtained via D2D communication. It can reduce the base station load and improve the caching hit rate. Compared to the other two caching strategies, the NIC mechanism can consider the physical intimacy between user devices and make the cache hit node closer to the request devices and obtain a higher data rate.

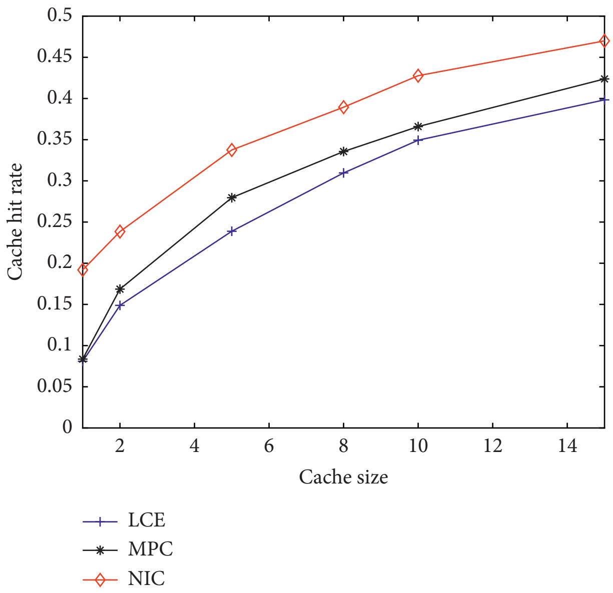


Figure 2: Influences of cache size on the cache hit rate.

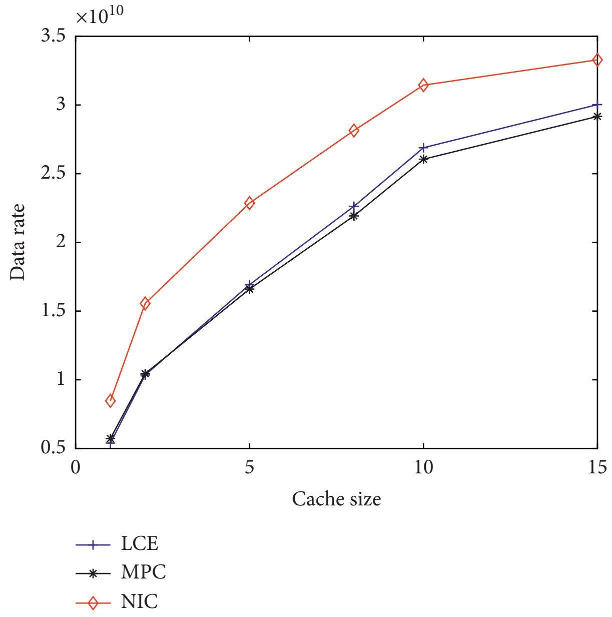


Figure 3: Influences of cache size on the data rate.

The influences of the Zipf parameters on the three caching mechanisms are shown in Figures 4 and 5. The bigger the Zipf parameter is, the more similar the user requests will be, and there will be a greater repeated request time for a small part of the files. As time elapses, the cached contents in the D2D network increase, and the cached contents are centralized in a small number of files. Then, users have a higher probability to receive files via the D2D communication link. Thus, the cache hit rate and data rate of the three caching mechanisms grow as Zipf parameters enlarge. The caching performance of NIC was always superior to the other two caching strategies, which was more significant when the Zipf parameter was smaller. The reason for this is that NIC can improve the diversity of the cached contents and improve caching performance by using network coding technology without increasing the cache size.

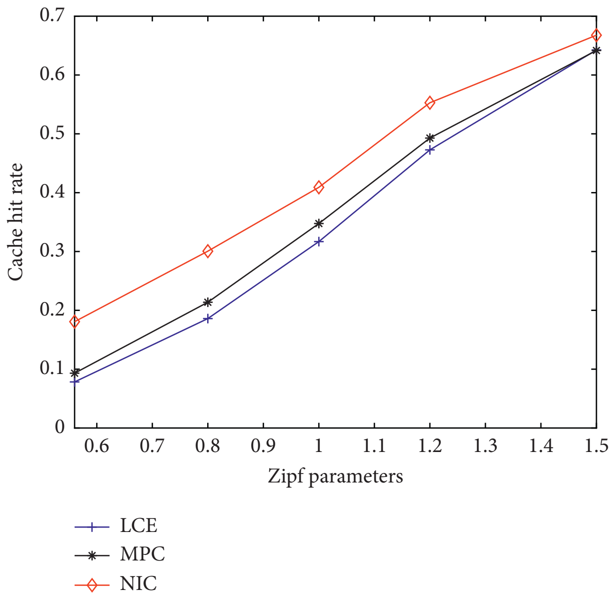


Figure 4: Influences of the Zipf parameters on the cache hit rate.

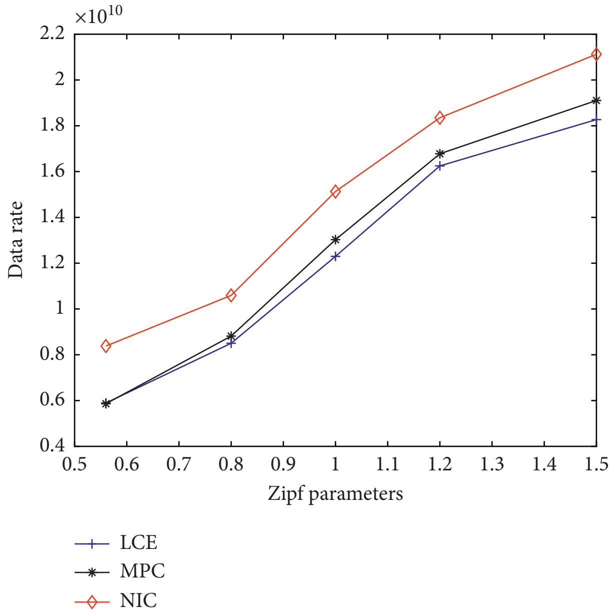


Figure 5: Influences of the Zipf parameters on the data rate.

5. Conclusion

In this paper, we propose a privacy-preserving D2D caching strategy based on the node importance, which can improve the diversity of the cached content by using the network coding technology and preserve the privacy of users and data. The SDN switch collects the information of requests and meeting information of the devices, and the SDN controller can compute the physical intimacy and request similarity between user devices and other devices by using the history information collected by SDN switches to obtain the node importance of the device. The node importance is defined as the weighted sum of the physical intimacy and request similarity. The nodes with higher importance have higher cache benefits and will cache the original blocks; the nodes with lower importance have lower cache benefits and will cache the coding blocks to make the cached contents be distributed more reasonably in the network. Base station will decide which device can establish reliable and secure communication with the requester based on historical information, which reflects the importance and social trust of devices. The simulation results show that the caching strategy based on node importance in this paper could improve the cache hit rate and data rate and effectively improve the performance and security of the caching network compared to the other two proactive and passive caches.

Data Availability

No data were used to support this study.

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 National Natural Science Foundation of China (62002072) and the Guangzhou Science and Technology Project (GZHKJXM20170117).

References

[1] C. Cisco, *Visual Networking Index: Global Mobile Data Traffic Forecast Update*, 2016 WhitePaper, German, Netherlands, 2017.

[2] L. Qiu and G. Cao, “Popularity-Aware caching increases the capacity of wireless networks,” *IEEE Transactions on Mobile Computing*, vol. 19, no. 1, pp. 173–187, 2020.

[3] Y. Zhou, L. Chen, C. Yang, and D. M. Chiu, “Video popularity dynamics and its implication for replication,” *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1273–1285, 2015.

[4] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, “A survey on low latency towards 5G: RAN, core network and caching solutions,” *IEEE Communications Surveys and Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018.

[5] K. Guo, C. Yang, and T. Liu, “Caching in base station with recommendation via Q-learning,” in *Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, San Francisco, CA, USA, March 2017.

[6] P. Blasco and D. Gündüz, “Learning-based optimization of cache content in a small cell base station,” in *Proceedings of the 2014 IEEE International Conference on Communications (ICC)*, pp. 1897–1903, Sydney, Australia, June 2014.

[7] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. Leung, “Cache in the air: exploiting content caching and delivery techniques for 5G systems,” *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131–139, 2014.

[8] Y. Wang, Y. Chen, H. Dai, Y. Huang, and L. Yang, “A learning-based approach for proactive caching in wireless communication networks,” in *Proceedings of the 2017 9th International Conference on Wireless Communications and*

Signal Processing (WCSP), pp. 1–6, Nanjing, China, October 2017.

[9] B. Bai, L. Wang, Z. Han, W. Chen, and T. Svensson, "Caching based socially-aware D2D communications in wireless content delivery networks: a hypergraph framework," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 74–81, 2016.

[10] M. Afshang, H. S. Dhillon, and P. H. J. Chong, "Fundamentals of cluster-centric content placement in cache-enabled device-to-device networks," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2511–2526, 2016.

[11] L. Xiao, X. Wan, C. Dai, X. Du, X. Chen, and M. Guizani, "Security in mobile edge caching with reinforcement learning," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 116–122, 2018.

[12] R. Ahlswede, N. Ning Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[13] J. Wang, J. Ren, K. Lu, J. Wang, S. Liu, and C. Westphal, "A minimum cost cache management framework for information-centric networks with network coding," *Computer Networks*, vol. 110, pp. 1–17, 2016.

[14] N. Golrezaei, P. Mansourifard, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3665–3676, 2014.

[15] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: a new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013.

[16] R. Wang, J. Zhang, S. H. Song, and K. B. Letaief, "Mobility-Aware caching in D2D networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 5001–5015, 2017.

[17] B. Chen, C. Yang, and A. F. Molisch, "Cache-Enabled device-to-device communications: offloading gain and energy cost," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4519–4536, 2017.

[18] D. Malak, M. Al-Shalash, and J. G. Andrews, "Spatially correlated content caching for device-to-device communications," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 56–70, 2018.

[19] K. Wu, M. Jiang, F. She, and X. Chen, "Relay-aided request-aware distributed packet caching for device-to-device communication," *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 217–220, 2019.

[20] W. Jiang, G. Feng, S. Qin, T. S. P. Yum, and G. Cao, "Multi-agent reinforcement learning for efficient content caching in mobile D2D networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 3, pp. 1610–1622, 2019.

[21] L. Li, Y. Xu, J. Yin et al., "Deep reinforcement learning approaches for content caching in cache-enabled D2D networks," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 544–557, 2020.

[22] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: the role of proactive caching in 5G wireless networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, 2014.

[23] B. Chen and C. Yang, "Caching policy optimization for D2D communications by learning user preference," in *Proceedings of the 2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pp. 1–6, Sydney, Australia, June 2017.

[24] Y. Zhang, E. Pan, L. Song, W. Saad, Z. Dawy, and Z. Han, "Social network aware device-to-device communication in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 177–190, 2015.

[25] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1801–1819, 2014.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 6654283, 11 pages
https://doi.org/10.1155/2020/6654283

Research Article

A Key Business Node Identification Model for Internet of Things Security

Lixia Xie,[1] **Huiyu Ni**,[1] **Hongyu Yang**,[1] **and Jiyong Zhang**[2]

[1] *School of Computer Science and Technology, Civil Aviation University of China, Tianjin 300300, China*
[2] *School of Computer and Communication Science, Swiss Federal Institute of Technology in Lausanne, CH-1015, Lausanne, Switzerland*

Correspondence should be addressed to Hongyu Yang; hyyang@cauc.edu.cn

Received 8 October 2020; Revised 25 October 2020; Accepted 1 December 2020; Published 12 December 2020

Academic Editor: Weizhi Meng

Copyright © 2020 Lixia Xie et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Based on the research of business continuity and information security of the Internet of Things (IoT), a key business node identification model for the Internet of Things security is proposed. First, the business nodes are obtained based on the business process, and the importance decision matrix of business nodes is constructed by quantifying the evaluation attributes of nodes. Second, the attribute weights are improved by the analytic hierarchy process (AHP) and entropy weighting method from subjective and objective dimensions to form the combination weight decision matrix, and the analytic hierarchy process and entropy weighting VIKOR (AE-VIKOR) method are used to calculate the business node importance coefficient to identify the key nodes. Finally, according to the NSL-KDD dataset, the network security events of IoT network intrusion detection based on machine learning are monitored purposefully, and after the information security event occurs in the smart mobile phone, which impacts through IoT on the business system, the impact of the key business node on business continuity is analyzed, and the business continuity risk value is calculated to evaluate the business risk to prove the effectiveness of the model. The experimental results of the civil aviation departure business show that the AE-VIKOR method can effectively identify key business node, and the impact of the key business node on business continuity is analyzed, which further proves the efficiency and accuracy of the model in identifying the key business node.

1. Introduction

Nowadays, with the rapid development of the Internet of Things, related research fields are more concerned about information security and business continuity. The Internet of things (IoT) and mobile technology [1] make multisystem cooperation more convenient, the multisystem cooperation is closely related to its business continuity. Therefore, due to the application of the IoT technology, when an information security event occurs [2], it may lead to delay or stagnation of business execution, which will inevitably affect business continuity. The security of the Internet of Things is one of the hotspots in various academic fields, such as information security and machine learning. In particular, machine learning is used for intrusion detection of the IoT. Belouch et al. [3] used a machine learning analysis framework to detect any anomalous events occurring in the network traffic flow. Liu et al. [4] examined specific attacks in the NSL-KDD dataset that can impact sensor nodes and networks in IoT settings and studied eleven machine learning algorithms to detect the introduced attacks. Xie et al. [5] designed a monitoring mechanism to detect link-flooding attack (LFA) based on the availability of the crucial links and trace route flows for IoT security. Yang et al. [6] proposed a malicious node detection model based on reputation with enhanced low energy adaptive clustering hierarchy (Enhanced LEACH) routing protocol for wireless network security.

Based on the management of business continuity, at present, there are many achievements in the research of business continuity security [7–13]. Key business node identification is very important for business recovery, which is one of the research hotspots in the field of risk assessment for the business process. Ali et al. [14] proposed a business continuity risk assessment framework for IoT services. Given the problems of information security risk assessment and business continuity management, Torabi et al. [15] put business continuity risk management into the framework of information security risk assessment through business continuity risk analysis. Belov et al. [16] proposed a risk value calculation of the business completion rate by studying the situation of the business resource completion rate and quantitatively assessed the business system risk. Hariyanti et al. [17] proposed a new information security risk assessment model based on the business process to improve the model based on the organization's assets. Silmie et al. [18] proposed a business continuity plan framework, which is a procedural guidance to create plans that prevent, prepare, respond, manage, and recover a business from any disruption. Diesch et al. [19] developed a comprehensive model of relevant management success factors for organizational information security to make appropriate decisions. The Vise Kriterijumska Optimizacija I Kompromisno Resenje in Serbian (VIKOR) method [20] is one of the common methods of multiattribute decision-making, which is often used in risk assessment, economics, management, and other hot fields. Yang et al. [21] proposed a hybrid multicriteria decision-making model based on the intuitionistic fuzzy number, extended Decision-Making Trial and Evaluation Laboratory (DEMATEL) method, and VIKOR algorithm to assess the information system security risk. Mohsen et al. [22] proposed an extended VIKOR method based on entropy measure for the failure modes of the geothermal power plant risk assessment. Han et al. [23] used the modified VIKOR method to identify and preferentially reinforce critical lines for skeleton-network of power systems. The Technique for Order Performance by Similarity to Ideal Solution (TOPSIS) method [24, 25] is also one of the classic multiattribute evaluation methods, which is compared with the method in this paper. In summary, the paper uses the AE-VIKOR method with combined weighting for eliminating the subjective influence of some attributes to identify effectively the key business node. The model analyzes the impact of key business nodes on business continuity and further proves the effectiveness of key identification.

The main contributions of this paper can be summarized as follows. A key business node identification model for Internet of Things security is proposed. The model in this paper identifies effectively the key business node and analyzes its impact on business continuity. The model is mainly focused on the following.

(1) The combined weighting from the subjective and objective dimensions is used to improve the attribute weights of the VIKOR method to identify the key business node. Compared with the single weighting method, such as the AHP method, the combined weighting makes the results more accurate, which is verified by experiments.

(2) After the information security event occurs in the smart mobile phone, which impacts through IoT on the business system, the model can be used to analyze the impact of the key business node on business continuity. For the specific business of the business process, this model analyzes the number of business users, business average execution time, and resource utilization.

(3) According to the business user number, business average execution time, and resource utilization, the business continuity risk value is calculated and realizes properly the risk assessment of business continuity in the model.

(4) In this model, the decision coefficient is selected reasonably by the experiment to realize accurate identification of key business node. Compared with other multiple attribute decision-making cases, such as using the VIKOR method to select coal suppliers, it is novel that the paper analyzed the influence of different decision mechanism coefficients on the identification results. After the key nodes are identified by this model, the key nodes are further analyzed to facilitate the analysis of the impact of business continuity.

The organization of this paper is described as follows. In section 2, a key business node identification model for the Internet of Things security is proposed. The key business node identification model is composed of four modules: data preparation module, data operation module, decision module, and analysis module. In section 3, the data preparation module and the data operation module are described in detail. The decision module and analysis module are expounded in section 4. In section 5, the effectiveness of the model is verified by analyzing the business continuity of the departure business and the loading business. Conclusion is given in section 6.

2. Key Business Node Identification Model

The key business node identification model is composed of four modules: data preparation module, data operation module, decision module, and analysis module. The framework diagram of the model is shown (see Figure 1).

The function design of each module in the model is as follows.

(1) Data preparation module: according to the business process, the business node set to be evaluated is obtained, and the node importance decision matrix is obtained from the business node set and the evaluation attribute.

(2) Data operation module: in this module, AHP subjective weighting and entropy objective weighting methods are used. The decision matrix is weighted by the combined weight from the subjective and objective dimensions, and the node importance combined weight decision matrix is formed.

(3) Decision module: in this module, the combination weights are used to improve the attribute weight of the VIKOR method to get the node importance

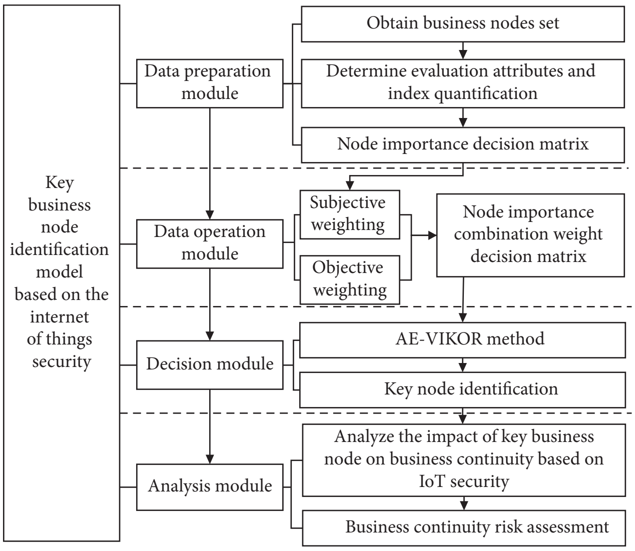

Key business node identification model based on the internet of things security
Data preparation module
Obtain business nodes set
Determine evaluation attributes and index quantification
Node importance decision matrix
Data operation module
Subjective weighting
Objective weighting
Node importance combination weight decision matrix
Decision module
AE-VIKOR method
Key node identification
Analysis module
Analyze the impact of key business node on business continuity based on IoT security
Business continuity risk assessment

FIGURE 1: Key business node identification model for business process.

coefficient and rank it. The key business node is identified in this model.

(4) Analysis module: when information security events occur, business continuity faces risks. The impact of key business nodes on business continuity is analyzed in this module, the business continuity risk value is calculated, and business continuity risk assessment is carried out.

3. Data Preparation Module and Data Operation Module

3.1. Data Preparation Module. IoT allows billions of devices as well as virtual environments to exchange data with each other intelligently. For example, smartphones have become an important personal assistant and indispensable part of people's everyday life and work. With such a large amount of data, the model first analyzes business processes to better analyze business continuity. Through the analysis of the business process, this model extracts all businesses into nodes to form the business node set to be evaluated, which is recorded as $M = \{n_1, n_2, n_3, \ldots, n_m\}$. M is the business node set to be evaluated. The set indicates that there are m nodes in the business process, which are numbered as $n_1, n_2, n_3, \ldots, n_m$.

Considering business importance from multiple perspectives makes the identification of key business more effective. Therefore, this paper selects three factors to evaluate business importance, which are business node relevance, business user, and business priority.

The specific process of indicator quantification of the business node importance attribute is as follows.

First, according to the theory of business process and complex network node centrality [26], business relevance is considered to assess business importance, and business relevance value can be measured according to the direct relationship between other business nodes and the business node. The value of business node relevance is calculated according to (1). The larger the business node relevance value is, the more important the business is:

$$g_i = \frac{h_i}{(m-1)}, \tag{1}$$

where g_i is the ratio of the number of connected nodes of business node i to the total number of nodes except for node i. The larger the value is, the more important the business node is. h_i is the number of nodes directly connected to node i. m is the total number of business nodes.

Second, the business user importance is used to evaluate business importance. The types of business users are divided into staff, ordinary users, and both staff and ordinary users. In this paper, levels one, two, and three are assigned to business user types.

Different types of users have different initial values. The larger the value is, the more important the business is. The importance levels for business user type values are defined in Table 1.

Finally, business priorities based on different business service types are used to evaluate business importance. The higher the business priority level, the higher the importance of business.

The business priority assignment is based on the service characteristics and application types of the business. The business priority level is divided into levels one, two, three, and four. The assignment is shown in Table 2.

The data preparation module forms the node importance decision matrix **X** through the quantification of attributes and the nodes obtained. Due to the different dimensions of each attribute, matrix **X** is normalized by (3) for comparison. The standardized matrix is written as **R**.

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & x_{m3} \end{bmatrix}, \tag{2}$$

$$r_{ij} = \frac{\left(x_{ij} - x_j^{\min}\right)}{\left(x_j^{\max} - x_j^{\min}\right)}, \tag{3}$$

where $x_j^{\max} = \max\{x_{i1}, x_{i2}, x_{i3}\}$ and $x_j^{\min} = \min\{x_{i1}, x_{i2}, x_{i3}\}$ in (3).

3.2. Data Operation Module. To eliminate some subjective influence of attributes and enhance the accuracy of the model, this paper uses the combined subjective and objective weighting method to determine the attribute weight.

The AHP method is one of the common methods to calculate the subjective weight. First, three attributes are compared. The business relevance is the local attribute of the business nodes, and its impact is relatively low. When the business user directly affects business operations, the impact of the user is stronger than that of the business relevance,

Table 1: Importance level of business user type.

Category	Value
Ordinary users	1
Staff member	2
Both staff and ordinary users	3

Table 2: Business priority assignment.

Business	Service	Application type	Business priority
Background	Without time delay	No special requirement for the business transmission time	1
Interactive	On demand response	Online data interaction of business characterized by the request response mode	2
Flow pattern	Time delay	Real-time business with low interaction	3
Conversation	Time delay strictly	Real-time business with high quality interaction	4

and the impact of the business type is greater than others. Therefore, the comparison of the attribute of node importance evaluation is shown in Table 3.

Table 3: Importance level of business user type.

Meaning	Value
Attribute i has the same effect as attribute j	1
Attribute i has a stronger influence than attribute j	3
Attribute i is an absolutely stronger influence than attribute j	5

where 2 and 4 indicate that the influence degree of attribute i and attribute j is between 3 and 5.

The subjective weighting steps are as follows.

Step 1: according to the subjective influence of business attributes on business importance, an initial comparison matrix **A** is constructed.

$$\mathbf{A} = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{5} \\ 3 & 1 & \frac{1}{3} \\ 5 & 3 & 1 \end{bmatrix}. \tag{4}$$

Matrix **A** is normalized to form matrix **B** according to the following:

$$\mathbf{B} = \frac{\mathbf{A}_{ij}}{\sum_{j=1}^{3} \mathbf{A}_{ij}}, \tag{5}$$

$$\mathbf{B} = \begin{bmatrix} \frac{1}{9} & \frac{1}{13} & \frac{1}{23} \\ \frac{1}{9} & \frac{3}{13} & \frac{5}{23} \\ \frac{1}{9} & \frac{9}{13} & \frac{13}{23} \end{bmatrix}. \tag{6}$$

Step 2: calculate the sum of each row of matrix **B** and get set **S** which is {0.3185, 0.7815, 1.9000}. The set is standardized to get the other set S_1 which is {0.1062, 0.2605, 0.6333}. The element of set S_1 is the subjective weight.

The AHP method coordinates the importance of each attribute to avoid the contradiction of each scheme. Therefore, it is necessary to meet the consistency test. After the consistency test, the calculation of consistency test index CI is shown as follows:

$$\mathrm{CI} = \frac{(\lambda_{\max} - n)}{(n-1)}, \tag{7}$$

$$\mathbf{A}\mathbf{W} = \lambda_{\max}\mathbf{W}, \tag{8}$$

where $\lambda_{\max}$ is the maximum eigenvalue and **W** is the maximum eigenvector in (8).

After testing, the subjective weight assignment conforms to the consistency test index. Therefore, the subjective weight of each attribute is obtained which are $w_1^A = 0.1062$, $w_2^A = 0.2065$, and $w_3^A = 0.6333$.

Entropy weighting is one of the classical methods to calculate objective weight. Using entropy value to modify the index weight provides a more reliable basis for the evaluation of business importance. The objective weight is calculated as follows:

$$S_{ij} = \frac{r_{ij}}{\sum_{i=1}^{m} r_{ij}}, \tag{9}$$

$$e_j = -k \sum_{j=1}^{n} S_{ij} \ln S_{ij}, \quad j = 1, 2, \ldots, n, \tag{10}$$

$$w_j = \frac{1 - e_j}{\sum_{j=1}^{n} 1 - e_j}, \tag{11}$$

where S_{ij} is the proportion of each indicator of each node in (9), and e_j is the information entropy of the j-th index. The objective weight of each attribute is obtained, which are defined as w_1^O, w_2^O, and w_3^O.

Combined weight combines subjective weight and objective weight. The weight matrix $\mathbf{Y}$ is constructed based on the subjective and the objective method. The combined weight of attributes is calculated by (9)–(11) which is defined as $w^z = (w_1^z, w_2^z, w_3^z)$:

$$\mathbf{Y} = \begin{bmatrix} w_1^A & w_1^O \\ w_2^A & w_2^O \\ w_3^A & w_3^O \end{bmatrix}, \tag{12}$$

$$\left[\left(\mathbf{R}^T\mathbf{Y}\right)^T\left(\mathbf{R}^T\mathbf{Y}\right)\right]\mathbf{X}^* = \lambda_{\max}\mathbf{X}^*, \tag{13}$$

$$\mathbf{W} = \mathbf{Y}\mathbf{X}^*, \tag{14}$$

$$w_i^z = \left(\frac{w_1^*}{\sum_{j=1}^3 w_j^*}, \frac{w_2^*}{\sum_{j=1}^3 w_j^*}, \frac{w_3^*}{\sum_{j=1}^3 w_j^*}\right), \tag{15}$$

$$\mathbf{C} = w_i^z \times \mathbf{R}, \tag{16}$$

where $\lambda_{\max}$and $\mathbf{X}^*$ are the largest eigenvalue and the largest eigenvector of R, respectively, in (13). The standardized decision matrix $\mathbf{C}$ of node importance combined weight is calculated by (16).

4. Decision Module and Analysis Module

4.1. Decision Module. The importance coefficient of business is calculated and sorted based on the AE-VIKOR method in the decision module. The AE-VIKOR method improves the evaluation attribute weight of the VIKOR method by combined weighting in the data operation module detailed in section 3.

VIKOR method is one of the common methods of the multiattribute decision model. The method considers both the maximum group utility and the minimum individual regret effect of the object; VIKOR method focuses on ranking and selecting from a set of alternatives and determines compromise solutions for a problem with conflicting criteria, which can help the decision-makers to reach a final decision.

The value of the maximum group utility is measured by U_i, the value of the minimum individual regret effect is expressed by K_i, and Q_i is the decision value, which is calculated by the following:

$$U_i = \sum_{i=1}^{3} w_i^z c_{ij}, \tag{17}$$

$$K_i = \max_i\left(w_i^z c_{ij}\right), \tag{18}$$

$$Q_i = v\frac{U_i - U^*}{U^- - U^*} + (1-v)\frac{K_i - K^*}{K^- - K^*}, \tag{19}$$

where v is the coefficient of the decision-making mechanism in (19), $U^* = \min_i U_i$, $U^- = \max_i U_i$, $K^* = \min_i K_i$, and $K^- = \max_i K_i$. Through comparative experimental analysis in section 5, in order not to lose the generality, this paper selects $v = 0.5$.

AE-VIKOR method is also a compromise ranking method, the feasible solution of which is closest to the ideal solution. Therefore, the AE-VIKOR method is without loss of generality to meet the following two conditions.

Condition 1. Acceptable advantage. The first two nodes in sorting are Q_i and Q_j. The conditions shown in formula (16) need to be met, where m is the number of business nodes.

$$Q_i - Q_j \geq \frac{1}{(m-1)}. \tag{20}$$

Condition 2. Acceptable stability. The importance coefficients of key business nodes rank first in U_i and K_i.

If the aforementioned two conditions are met at the same time, the model recognition results are considered valid. The value of Q_i calculated based on the AE-VIKOR method is the business importance coefficient. The key business node is the largest business importance coefficient. Through the calculation of the AE-VIKOR method, the business importance coefficient is between [0, 1].

4.2. Analysis Module. The information security of IoT is closely related to business continuity management in the Internet era. When an information security event occurs in the system, it will affect the business continuity for the business process.

When a threat makes use of the vulnerability of IoT, information security events will appear, such as natural disaster events, infrastructure failures, network attacks, technical failures, and malicious code attacks. Therefore, it shows the relationship between information security and business continuity (see Figure 2).

The risk value of business continuity is calculated by combining the importance coefficient of key business according to the number of business users, average execution time of business, and resource utilization in this paper.

In this paper, the maximum of business user's numbers, average execution time, and resource utilization are, respectively, set as $u_{\max}, r_{\max}, t_{\max}$. When an information security event occurs, the number of business users, business execution time, and resource utilization rate at i time are defined as u_i, r_i, t_i. The business continuity risk value is calculated by the following:

$$P_i = 1 - \left(\frac{1}{3}\right)\frac{\sum(u_i, r_i, t_i)}{(u_{\max}, r_{\max}, t_{\max})}, \tag{21}$$

$$\Delta P = P_1 - P_2, \tag{22}$$

$$L = Q_i * \Delta P, \tag{23}$$

where Q_i represents the business importance coefficient, which can be calculated by the AE-VIKOR method in section 3. L represents the business continuity risk value, which is an important basis for the business continuity risk assessment level.

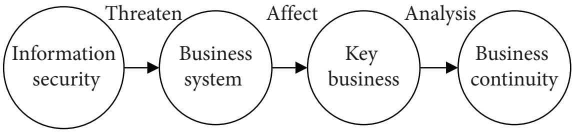


Figure 2: Relationship between information security and business continuity.

On calculating according to (21)–(23), the business continuity risk value L is an important basis for the business continuity risk assessment level. Because the value range ΔP is between 0 and 1, and the business importance coefficient is between 0 and 1, business continuity risk is classified according to business continuity risk value. When the risk value of business continuity is higher than 0.15, it is considered that business continuity is at higher risk. Business continuity changes with the change of business execution time. The experimental results based on mobile devices show that, after the completion of service execution time, the business continuity risk value calculated by the model does not exceed 0.15. Therefore, the use of academic language to describe business continuity risk is shown in Table 4. The business risk value is between 0 and 0.15, so the risk level of business continuity is shown in Table 4.

Table 4: The risk level of business continuity.

Business continuity risk value	Business continuity risk level
0~0.05	Low
0.05~0.10	Medium
0.10~0.15	High
$L \geq 0.15$	Higher risk

5. Experimental Results and Analysis

The civil aviation industry is one of the key industries of information security. Due to the convenience of IoT, it is very common for the public to handle the departure business on mobile devices. In particular, the check-in service is carried out through the IoT technology on the smart mobile devices. However, information security appears in the smart mobile devices, and other services connected through IoT technology will also be affected. As one of the core business systems in the field of civil aviation, departure system security is of great significance. To ensure the operation safety of the civil aviation business, this paper studies the potential security risks and possible risks of civil aviation information. Therefore, it is of great significance to analyze the implementation and business continuity of the key services of mobile devices.

5.1. Key Node Identification. According to the NSL-KDD dataset, the network security events of IoT network intrusion detection based on machine learning are monitored purposefully, and the risk of business continuity caused by the key business is analyzed. Once the information security event occurs in the smart mobile phone, which impacts through IoT on every business of the system, it will cause a great threat to civil aviation security.

Therefore, the experimental object of this paper is the departure business process of civil aviation. Its business process is shown (see Figure 3). Specific experimental steps of calculating the business importance coefficient are as follows.

Step 1. Obtain the business node set.

This experiment needs to evaluate the importance of all business nodes in the departure business process. Therefore, all businesses in the departure business process are extracted into nodes to form the business node set to be evaluated, which is recorded as $\mathbf{N} = \{n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9\}$, and represents establish flight information, flight data control prepared flight, loading preparation flight, check-in, monitoring the check-in flights, end check-in, end monitoring, loading closure, and flight closure, respectively.

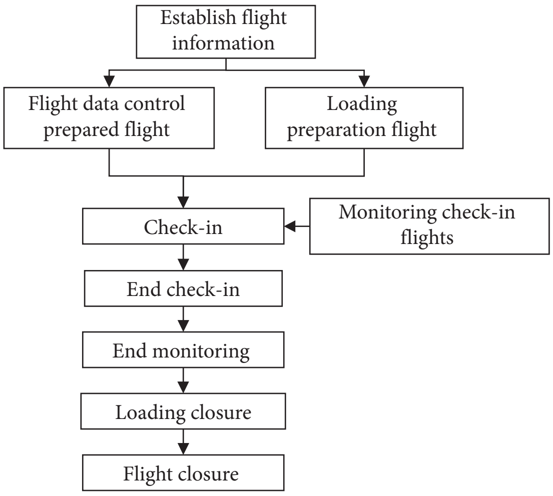

Establish flight information
Flight data control prepared flight
Loading preparation flight
Check-in
Monitoring check-in flights
End check-in
End monitoring
Loading closure
Flight closure

Figure 3: Framework diagram of departure business.

Step 2. Construct the decision matrix of node importance.

The decision matrix of node importance is formed by the node and the attributes of each node. According to the assignment of node attribute indicators in the data preparation module, the assignment of departure business node importance attribute indicators is shown in Table 5.

The node importance decision matrix $\mathbf{X}$ is formed according to the business nodes and quantitative values of each attribute shown in Table 5.

After (3) is standardized, the standardized node importance decision matrix $\mathbf{R}$ is formed:

$$\mathbf{R} = \begin{bmatrix} 0.2917 & 0.4330 & 0.1961 \\ 0.4376 & 0.2887 & 0.1961 \\ 0.2917 & 0.4330 & 0.3922 \\ 0.5835 & 0.4330 & 0.5883 \\ 0.1459 & 0.2887 & 0.3922 \\ 0.2917 & 0.2887 & 0.1961 \\ 0.1459 & 0.2887 & 0.1961 \\ 0.2917 & 01443 & 0.1961 \\ 0.2917 & 0.2887 & 0.1961 \end{bmatrix}. \tag{24}$$

Table 5: Assignment of the important attribute index of departure business nodes.

Node	Business relevance	Business user	Business priority
n_1	0.2500	3	1
n_2	0.3750	2	1
n_3	0.2500	3	2
n_4	0.5000	3	3
n_5	0.1250	2	2
n_6	0.2500	2	2
n_7	0.1250	2	1
n_8	0.2500	1	1
n_9	0.2500	2	1

Step 3. Calculate the combined weight.

The combined weight is calculated. The subjective weight is $w^A = \{0.1062, 0.2605, 0.6333\}$, which is calculated by the AHP method. According to the objective weight calculated by the entropy method, $w^O = \{0.3273, 0.3298, 0.3429\}$ according to (9)–(11), and the combined weight of the two is $w^Z = \{0.2228, 0.2756, 0.5016\}$ according to (13)–(16).

Step 4. Key business node identification.

Node importance ranking based on the AE-VIKOR method in section 4 and the importance coefficient of departure business node are calculated as $Q_i = \{0.1975, 0.1464, 0.5199, 1.000, 0.4844, 0.4947, 0.1048, 0.057, 0.1176\}$ according to (13)–(15). The recognition results of the model meet two conditions after testing according to (16)–(19), and then the identification result of the key node identification model is regarded as valid. It can be seen that the most important factor of n_4 is the node. It is the check-in business that is the key business of the departure system.

5.2. Business Continuity Analysis. When a threat makes use of the vulnerability of IoT, an information security event occurs in the passenger check-in system, and the maximum number of business users, average execution time, and resource utilization rate of the passenger check-in system at T_0 time, respectively, correspond to 1000, 10 s, and 90%.

After the information security event occurs in the check-in system at T_0 time, the check-in system data within 1 h can be obtained through monitoring. Table 6 shows the execution of the check-in business at T_1, T_2, T_3, T_4 after the information security event.

To compare with the check-in, when the information security event occurs in the loading system at the time, the system data within 1 h is monitored and obtained. Problems in the loading system affected the loading fight business and loading closure business.

Execution of the loading preparation business after the information security event is shown in Table 7. The execution of the loading closure business after the information security event is shown in Table 8.

The data in Tables 6–8 show, after the occurrence of information security incidents, the three factors related to business continuity, namely, the number of business users, average execution time of business, and change of resource utilization rate with time.

The time of the loading system is inconsistent with the time of the aforementioned passenger check-in system, and the time of information security incident is inconsistent, while the monitoring time and time interval are consistent. Therefore, the business continuity risk value and assessment level are shown in Figure 4 at the same time.

The data of check-in business and loading business at every moment shows the degree of business continuity risk in Figure 4.

When an information security event occurs at T_0 time, it can be seen from Figure 4 that the business continuity risk value of check-in business increases rapidly after the time, while that of the loading fight business is relatively slow compared with check-in business. The data of the loading closure business shows it has the least impact on business continuity and the change degree of business continuity risk of the loading closure is the least.

At T_4 time, the value of the business continuity risk of the check-in business is 0.1426, and it is close to the higher risk. The data shows that the business continuity risk value of loading fights business within T_4 time is slowly increasing, and the risk of the loading fights business at T_4 time is 0.0654, and the corresponding risk level of business continuity is medium. At T_4 time, the risk of the loading closure business is 0.0086. Its risk increases more slowly with the change of time. The corresponding risk level of business continuity is low at T_4 time.

Therefore, the experiment further proves the validity and accuracy of the key business node identification model based on the AE-VIKOR method, and the impact of key nodes on business continuity is clearly demonstrated in Figure 4.

5.3. Comparison of Key Business Identification Methods. In this paper, the AE-VIKOR method is used to calculate the importance coefficient of civil aviation departure business nodes, and the AE-VIKOR method is compared with the other five methods. The importance coefficient calculated by each method for each node is shown in Table 9. The calculation method and business node ranking of several business nodes are shown (see Figure 5) to clearly describe the difference between each method. Therefore, the value in Figure 5 corresponds to the importance coefficient calculated in Table 9.

As can be seen from Figure 5, the AE-VIKOR method is more accurate than the other four methods. AHP-VIKOR and Entropy-VIKOR methods consider attribute weight from a single perspective, and then, the evaluation results from subjective or objective perspectives are biased. The VIKOR method does not consider attribute weight and it is not an accurate assessment of business importance from multiple perspectives. The DEMATEL and AHP methods are used to calculate the subjective weight. By comparison, the weight calculated by the AHP method is better than that

Table 6: Execution of the check-in business after an information security event.

Time	Number of business users	Business average execution time/s	Resource utilization
T_0	1000	10.0	90
T_1	800	10.5	85
T_2	550	12.0	65
T_3	300	12.5	50
T_4	100	13.0	25

Table 7: Execution of the loading fight business after an information security event.

Time	Number of business users	Business average execution time/s	Resource utilization
T_0	100	5.0	98
T_1	80	6.5	81
T_2	55	7.2	69
T_3	30	7.8	50
T_4	10	8.0	28

Table 8: Execution of the loading closure business after an information security event.

Time	Number of business users	Business average execution time/s	Resource utilization (%)
T_0	800	3.0	95
T_1	675	4.0	72
T_2	574	4.3	63
T_3	350	4.8	57
T_4	190	5.0	26

by the DEMATEL method. For example, there is not much difference between the value of n_1 and that of n_2 calculated by the DEMATEL-Entropy-VIKOR method, and the difference in importance is not clearly expressed. However, the AE-VIKOR method clearly shows the difference in importance coefficients between the two nodes.

In this paper, the combined weight is applied to the TOPSIS method and compared with the AE-VIKOR method. The results show that the business importance coefficients of n_1, n_3, n_5 calculated by the TOPSIS method are also biased compared with the AE-VIKOR method (see Figure 5). Therefore, this paper uses the AHP method to calculate the subjective weight of attributes and uses the entropy method to calculate objective weights. From these two dimensions, the combined weights are considered to improve the attribute weights of the VIKOR method and further improve the model recognition effect. This paper uses the AE-VIKOR method to calculate the business importance coefficient to ensure the accuracy of the results to facilitate the analysis and management of business continuity.

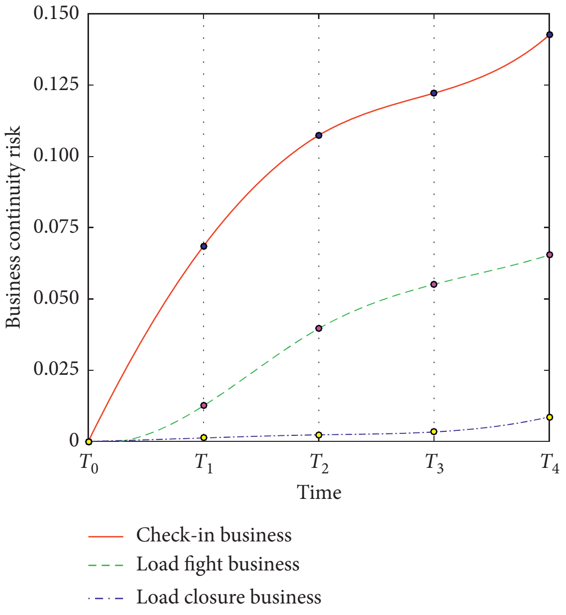


Figure 4: Business continuity risk analysis.

5.4. Comparative Experiment on the Coefficient Selection of Decision Mechanism.

The evaluation results of the AE-VIKOR method are different due to different coefficients of decision mechanism v. It is very important to choose the coefficient of decision mechanism reasonably for the evaluation result of the method. To adopt a reasonable and efficient decision mechanism, coefficient v is designed to be 0.2, 0.4, 0.5, 0.6, and 0.8, in this paper. The importance coefficient of departure business node is calculated and analyzed. The evaluation result is shown (see Figure 6).

It can be seen from Figure 6 that when $v = 0.5$, the calculation of the importance coefficient of each node is accurate and the difference is obvious. Therefore, to improve the universality of the model. The decision mechanism coefficient v of the AE-VIKOR method is set to 0.5.

Table 9: The importance coefficient for each node.

	AHP-VIKOR	Entropy-VIKOR	DEMATEL-Entropy-VIKOR	VIKOR	TOPSIS	AE-VIKOR
n_1	0.0879	0.3721	0.3623	0.3897	0.1697	0.1974
n_2	0.0668	0.3721	0.3748	0.3982	0.1058	0.1464
n_3	0.5211	0.4732	0.4689	0.4904	0.5106	0.5199
n_4	1.0000	1.0000	0.7645	1.0000	1.0000	1.0000
n_5	0.4543	0.2877	0.2777	0.2734	0.4047	0.4844
n_6	0.4772	0.3611	0.4611	0.3483	0.4523	0.4949
n_7	0.0211	0.0001	0.0001	0.0001	0.0106	0.1048
n_8	0.0001	0.0015	0.0018	0.0058	0.0001	0.0567
n_9	0.0439	0.0746	0.0846	0.0799	0.0582	0.1175

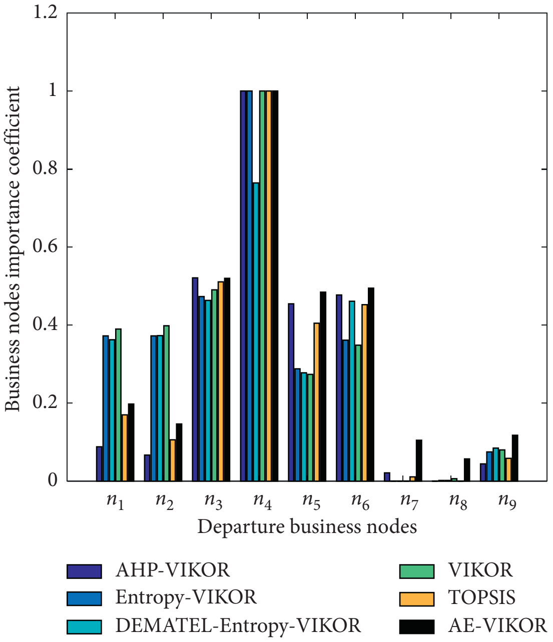


Figure 5: Business nodes importance coefficient calculated by different methods.

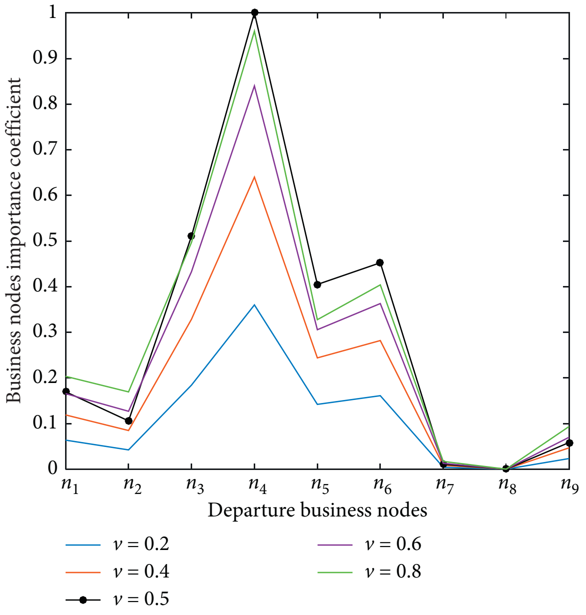


Figure 6: Comparison of different decision mechanism coefficient selection experiments.

6. Conclusion

This paper proposes a key business node identification model for the Internet of Things security. The model analyzed the business process to obtain business nodes. Then the business node importance evaluation attributes were quantified. And a combined weight was used to improve the attribute weight to identify key business node. After the information security event occurs in the smart mobile phone which impacts through IoT on the business system, the AE-VIKOR method is used to make a decision and sort the importance of business nodes, and the model analyzes the impact of key business node' on business continuity. The experimental results show that the key business node identification model based on the AE-VIKOR method is more accurate, and the business continuity risk assessment is carried out reasonably. The next step is to analyze the impact of the key business node on business recovery priority, after information security events occur, and further improve the recognition ability and adaptive ability of the model.

Data Availability

The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Disclosure

The manuscript has been extended about 150% from the Frontiers in Cyber Security (FCS 2020) conference manuscript. An earlier version of this work was presented as a paper at the FCS 2020 conference found at the following link: https://link.springer.com/chapter/10.1007/978-981-15-9739-8_47.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper when handling and making decisions.

Acknowledgments

This work was supported by the Civil Aviation Joint Research Fund Project of the National Natural Science Foundation of China under grant number U1833107.

References

[1] C. Fang, J. Liu, and Z. Lei, "Fine-grained HTTP web traffic analysis based on large-scale mobile datasets," *IEEE Access*, vol. 4, pp. 4364–4373, 2016.

[2] A. Jurcut, T. Niculcea, P. Ranaweera, and N.-A. Le-Khac, "Security considerations for Internet of things: a survey," *SN Computer Science*, vol. 1, no. 4, p. 193, 2020.

[3] M. Belouch, S. El Hadaj, and M. Idhammad, "Performance evaluation of intrusion detection based on machine learning using apache spark," *Procedia Computer Science*, vol. 127, pp. 1–6, 2018.

[4] J. Liu, B. Kantarci, and C. Adams, "Machine learning-driven intrusion detection for contiki-NG-based IoT networks exposed to NSL-KDD dataset," in *Proceedings of the ACM Workshop on Wireless Security and Machine Learning (WiseML 2020)*, Linz, Austria, July 2020.

[5] L. Xie, Y. Ding, H. Yang, and Z. Hu, "Mitigating LFA through segment rerouting in IoT environment with traceroute flow abnormality detection," *Journal of Network and Computer Applications*, vol. 164, Article ID 102690, 2020.

[6] H. Yang, X. Zhang, and F. Cheng, "A novel algorithm for improving malicious node detection effect in wireless sensor networks," *Mobile Networks and Applications*, pp. 1–10, 2020.

[7] A. N. Moldagulova, R. K. Uskenbayeva, R. Z. Satybaldiyeva et al., "On identification of hybrid business processes for effective implementation in the form of cloud services," in *Proceedings of the 2019 19th International Conference on Control, Automation and Systems (ICCAS)*, pp. 51–54, Jeju, Korea, October 2019.

[8] G. Sherzod, G. Abdukhalil, and V. Viktoriya, "Formalization of the business process security," in *Proceedings of the 2019 International Conference on Information Science and Communications Technologies (ICISCT)*, pp. 1–3, IEEE, Tashkent, Uzbekistan, November 2019.

[9] Q. Ming and L. Songtao, "Overview of system wide information management and security analysis," in *Proceedings of the 2017 IEEE 13th International Symposium on Autonomous Decentralized System (ISADS)*, pp. 191–194, IEEE, Bangkok, Thailand, March 2017.

[10] G. Stergiopoulos, P. Dedousis, and D. Gritzalis, "Automatic network restructuring and risk mitigation through business process asset dependency analysis," *Computers & Security*, vol. 96, Article ID 101869, 2020.

[11] R. Matulevičius, A. Norta, and S. Samarütel, "Security requirements elicitation from airline turnaround processes," *Business & Information Systems Engineering*, vol. 60, no. 4, pp. 3–20, 2018.

[12] H. Yang and G. Qin, "Identification of key systems for risk assessment," *Journal of Dalian University of Technology*, vol. 60, pp. 306–316, 2020.

[13] J. Xing, Z. Zeng, and E. Zio, "Dynamic business continuity assessment using condition monitoring data," *International Journal of Disaster Risk Reduction*, vol. 41, Article ID 101334, 2019.

[14] J. A. Ali, Q. Nasir, and F. T. Dweiri, "Business continuity framework for internet of things (IoT) services," *International Journal of System Assurance Engineering and Management*, vol. 11, pp. 1380–1394, 2020.

[15] S. A. Torabi, R. Giahi, and N. Sahebjamnia, "An enhanced risk assessment framework for business continuity management systems," *Safety Science*, vol. 89, pp. 201–218, 2016.

[16] V. M. Belov, A. I. Pestunov, and T. M. Pestunova, "On the issue of information security risks assessment of business processes," in *Proceedings of the 14th International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering APEIE*, pp. 136–139, IEEE, Piscataway, NJ, USA, October 2018.

[17] E. Hariyanti, A. Djunaidy, and D. O. Siahaan, "A conceptual model for information security risk considering business process perspective," in *Proceedings of the 2018 4th International Conference on Science and Technology (ICST)*, pp. 1–6, IEEE, Yogyakarta, Indonesia, August 2018.

[18] S. V. Fani and A. P. Subriadi, "Business continuity plan: examining of multi-usable framework," *Procedia Computer Science*, vol. 161, pp. 275–282, 2019.

[19] R. Diesch, M. Pfaff, and H. Krcmar, "A comprehensive model of information security factors for decision-makers," *Computers & Security*, vol. 92, Article ID 101747, 2020.

[20] D. Siregar, H. Nurdiyanto, S. Sriadhi et al., "Multi-attribute decision making with VIKOR method for any purpose decision," *Journal of Physics: Conference Series*, vol. 1019, 2018.

[21] J. Yang, J. Han, and X. Zhang, "Information system security risk assessment based on IDAV multi-criteria decision model," in *Proceedings of the 2018 12th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, pp. 121–127, IEEE, Xiamen, China, November 2018.

[22] O. Mohsen and N. Fereshteh, "An extended VIKOR method based on entropy measure for the failure modes risk assessment-a case study of the geothermal power plant (GPP)," *Safety Science*, vol. 92, pp. 160–172, 2017.

[23] C. Han, Y. Zhao, Z. Lin et al., "Critical lines identification for skeleton-network of power systems under extreme weather conditions based on the modified VIKOR method," *Energies*, vol. 11, 2018.

[24] P. Mateusz, M. Danuta, Ł. Małgorzata, B. Mariusz, and N. Kesra, "TOPSIS and VIKOR methods in study of sustainable development in the EU countries," *Procedia Computer Science*, vol. 126, pp. 1683–1692, 2018.

[25] Z. Wu, J. Xu, X. Jiang, and L. Zhong, "Two MAGDM models based on hesitant fuzzy linguistic term sets with possibility distributions: VIKOR and TOPSIS," *Information Sciences*, vol. 473, pp. 101–120, 2019.

[26] Y. Shen, C. Gu, and P. Zhao, "Structural vulnerability assessment of multi-energy system using a PageRank algorithm," *Energy Procedia*, vol. 158, pp. 6466–6471, 2019.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 6656066, 11 pages
https://doi.org/10.1155/2020/6656066

Research Article

An Unsupervised Learning-Based Network Threat Situation Assessment Model for Internet of Things

Hongyu Yang,[1] **Renyun Zeng**,[1] **Fengyan Wang**,[1] **Guangquan Xu**,[2] **and Jiyong Zhang**[3]

[1] *School of Computer Science and Technology, Civil Aviation University of China, 300300 Tianjin, China*
[2] *College of Intelligence and Computing, Tianjin University, 300350 Tianjin, China*
[3] *School of Computer and Communication Science, Swiss Federal Institute of Technology in Lausanne, CH-1015 Lausanne, Switzerland*

Correspondence should be addressed to Hongyu Yang; hyyang@cauc.edu.cn

Received 16 October 2020; Revised 28 October 2020; Accepted 16 November 2020; Published 28 November 2020

Academic Editor: Wenjuan Li

Copyright © 2020 Hongyu Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the wide application of network technology, the Internet of Things (IoT) systems are facing the increasingly serious situation of network threats; the network threat situation assessment becomes an important approach to solve these problems. Aiming at the traditional methods based on data category tag that has high modeling cost and low efficiency in the network threat situation assessment, this paper proposes a network threat situation assessment model based on unsupervised learning for IoT. Firstly, we combine the encoder of variational autoencoder (VAE) and the discriminator of generative adversarial networks (GAN) to form the V-G network. Then, we obtain the reconstruction error of each layer network by training the network collection layer of the V-G network with normal network traffic. Besides, we conduct the reconstruction error learning by the 3-layer variational autoencoder of the output layer and calculate the abnormal threshold of the training. Moreover, we carry out the group threat testing with the test dataset containing abnormal network traffic and calculate the threat probability of each test group. Finally, we obtain the threat situation value (TSV) according to the threat probability and the threat impact. The simulation results show that, compared with the other methods, this proposed method can evaluate the overall situation of network security threat more intuitively and has a stronger characterization ability for network threats.

1. Introduction

In recent years, the application of various emerging network technologies such as big data, blockchain, artificial intelligence, and other technologies in the field of Internet of Things (IoT) has brought about more and more convenience to people in many fields. At the same time, because of the connection with the Internet, the IoT devices are also vulnerable to more network threats [1], which will result in malicious attacks on physical devices. Reference [2] indicated that cyberphysical systems (CPSs) are vulnerable to traditional network threats, so the entire IoT system and the security and privacy of users are facing a huge threat. IoT devices and applications play an increasingly important role in critical infrastructure and everyday life; recent security incidents show that any successful attack will seriously hinder economic development and even endanger the safety of human life.

Because the IoT devices and applications are connected to the Internet, they are vulnerable to a variety of network attacks, which leads to important information leakage and even allows attackers to obtain permission to operate these devices. The authors of [3, 4] applied encryption algorithm in oblivious RAM to ensure the information security of storage devices. The IoT devices that are attacked by the network may have the management rights of the database stolen. To ensure the privacy and security of the database, the authors of [5, 6] proposed encryption algorithms to prevent the leakage of important information. However, in the face of a large number of complex network attacks, it is necessary to ensure network information security from a more comprehensive perspective.

To strengthen the construction of the network security defense system and deal with the emerging new threat attacks in the IoT network environment effectively, the stable and efficient network threat situation assessment (NTSA) method has become an important research topic. The NTSA evaluates the whole degree of security threats suffered by the IoT network system to analyze the situation of network attack and master the overall security situation of the network. NTSA can evaluate the current network security situation for IoT from a more comprehensive perspective and provide reliable information for network managers to make decision analysis and to minimize the loss that is caused by network threats [7]. However, in the past several years, the network has faced a large number of multisource threat attacks, which poses a huge threat to individuals and enterprises. The traditional network threat situation assessment method has the shortcomings of high modeling cost, low efficiency, and long cycle, which cannot make real-time and effective network security situation assessment.

To evaluate the network threat situation effectively in a multisource data environment of IoT, this paper proposes an unsupervised learning-based network threat situation assessment model for IoT. The contributions of this paper are as follows:

(1) To reduce the damage of network threats to IoT applications and devices, an unsupervised learning-based network threat situation assessment model was proposed. This model can reflect the current network situation of IoT effectively and provide decision support to network managers.

(2) This paper selects multisource heterogeneous network threat data to simulate the threats that IoT will be confronted with and calculate the threat situation value for the network threat situation assessment of IoT.

(3) The simulation results show that, compared to traditional models, this proposed method can evaluate the overall situation of network threats more intuitively and effectively for IoT.

1.1. Organization. The remainder of this paper is organized as follows. In Section 2, we present related works. Section 3 describes our proposed unsupervised network in detail. In Section 4, we propose our network threat situation assessment framework and the quantitative assessment process of the network threat situation in detail. Section 5 reports the experiments and the comparisons with other methods and, in the end, the conclusion is placed in Section 6.

2. Related Works

Assessment methods based on the mathematical model as applied to one of the earliest methods in network threat situation assessment and on account of its features such as being simple and easy to implement are widely used. Yang et al. [8] proposed a cloud computing risk assessment model that used the Markov chain (MC) model to describe the random risk environment and measured the risk value through information entropy (IE). Wang et al. [9] combined the analytic hierarchy process (AHP) with the hierarchical model of situational assessment and integrated the fuzzy results of multisource equipment with D-S evidence theory to solve the problem of single information source and large deviation of accuracy. Because the evaluation method based on the mathematical model is greatly influenced by subjective factors and there is no objective and unified standard definition variable, it is usually unable to achieve relatively perfect evaluation results.

Assessment methods based on probability and knowledge reasoning usually take advantage of the statistical characteristics of prior knowledge and combine with expert knowledge and experience database to build a model and then evaluate the threat situation by adopting logical reasoning. Sallam [10] identified potential network threats through fuzzy logic technology based on fuzzy reasoning (FR) engine and evaluated network security risks according to the attacker's overall capability, the overall probability of attack success, and the impact of the attack on three subfuzzy reasoning systems. Wen et al. [11] conducted a quantitative assessment of network security situation by fusing information sources with graded Naive Bayes classifier. These methods fuse various security assessment indicators in combination with the characteristics of mathematical statistics. However, the limitations of these methods are that they cannot give timely feedback and cannot meet the needs of task processing which result in a decrease in evaluation efficiency.

Deep-learning-based evaluation methods have been widely used in recent years because of their high efficiency and easy implementation. Feng et al. [12] extracted internal and external information features from the original time series network data and then trained and verified the extracted features in the recursive neural network (RNN) model, which has high predictive accuracy and robustness. He et al. [13] combined the wavelet neural network (WNN) with the maximum overlap discrete wavelet transform (MODWT) and proposed the network security situation prediction model through the data-driven method. Nevertheless, in the face of massive network security data, due to the lack of sufficient prior knowledge and established criteria of data category annotation, the task of manual category annotation is large and the cost is high, so the supervised data modeling method based on data label is gradually unable to apply to specific network scenarios.

Unsupervised learning (UL) provides an idea to solve the shortcomings of the above methods. Its main feature is that there is no need to label data categories manually but to conduct feature learning and modeling on the preprocessed data directly.

To evaluate the network threat situation of IoT effectively in a multisource data environment, this paper proposes a network threat situation assessment model based on unsupervised learning for IoT. It applies variant autoencoder and generative adversarial networks (V-G) model for cluster analysis of the training set; then the error threshold is calculated by the 3-layer variation automatic encoder. Then it uses the abnormal traffic datasets to conduct threat tests and quantify the network situation assessment according to the calculated results of the threat situation value. The experimental results show that the method presented in this paper has a good evaluation effect on network threats and has a strong characterization ability in the face of network threats. Furthermore, it can evaluate the network threat situation effectively without relying on data labels.

3. Unsupervised Generation Network Model

3.1. Variational Autoencoder (VAE) and Generative Adversarial Network (GAN). Autoencoder (AE) and variational autoencoder (VAE) [14] are both composed of encoder and decoder; the biggest difference between them is that VAE adds the "noise constraint" that compels the encoder to produce a collection of latent variables (LV) that are subject to the unit Gaussian distribution. The network structures of AE and VAE are demonstrated in Figures 1 and 2.

Comparing Figures 1 and 2, VAE compels every sample X_k in the original sample $X = \{X_1, X_2, X_3, ..., X_n\}$ to follow the normal distribution N (μ, σ^2), which means fitting the average μ and the variance σ^2 of any sample X_k by the internal neural network, and then obtains a set of potential variables $Z = \{Z_1, Z_2, Z_3, ..., Z_n\}$, in which the element Z_k is subject to the multivariate standard normal distribution $N(0, I)$. In the decoding process, Z generates the sample set $Y = \{Y_1, Y_2, Y_3, ..., Y_n\}$ through the decoder; then the similarity between the generated sample set Y and the original sample set X is statistically computed by the distance function. The reconstruction error loss of the overall data element can be obtained by calculation.

Generative adversarial network (GAN) [15] is one of the most promising deep generation network models in the field of unsupervised learning, which consists of a generator and a discriminator. The network structure of GAN is shown in Figure 3.

As shown in Figure 3, the generator first learns the probability distribution characteristics of a collection of random noises obtained by direct sampling through a prior distribution. Then it tries to generate the data sample $Y = \{Y_1, Y_2, Y_3, ..., Y_n\}$ which is the same as the original sample $X = \{X_1, X_2, X_3, ..., X_n\}$ to "trick" the discriminator that is responsible for determining the similarity between the generated sample Y and the original sample X. The output of the discriminator is a scalar in the range of [0, 1] for each similarity test. The closer the scalar gets to 0, the less likely the generated sample Y_k will be judged as real data. The closer the scalar gets to 1, the more likely the generated sample Y_k will be judged as real data.

Generator and discriminator compose a dynamic game process, and the generator is gradually acquiring the distribution features of the data after the repeated game; when the discriminator's output reaches the NASH equilibrium (NASH = 0.5), it can generate sample Y that has a high degree of similarity to the original sample X through a random noise Z. The training will finish when the discriminant is unable to distinguish between real data and generated data.

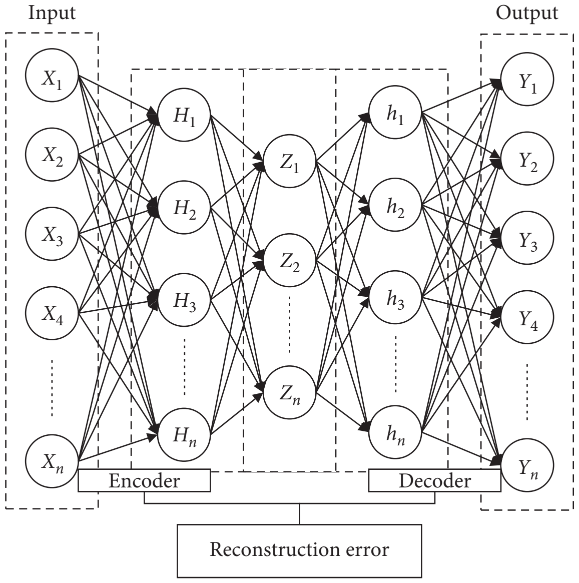


Figure 1: AE's network structure.

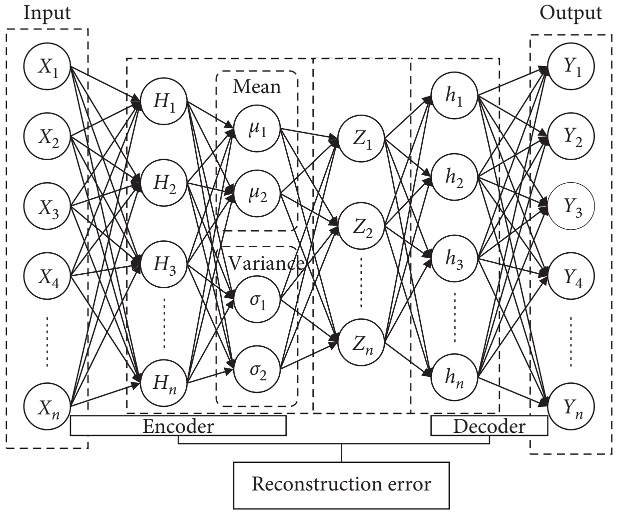


Figure 2: VAE's network structure.

3.2. V-G Network. The design of the V-G network is based on the following analysis:

(1) VAE can learn in the process of encoding data prior distribution and generate samples with good diversity performance while measuring the similarity between generated samples and original samples, can only use the mean square error (MSE) functions to roughly calculate the similarity errors between data elements, and is unable to adopt a more reasonable strategy of the similarity measure, which reduces the accuracy of matching samples.

(2) GAN has a high discriminant standard for generating samples and original samples when it judges the similarity of samples through discriminator. However, it is difficult for the fitting of real sample

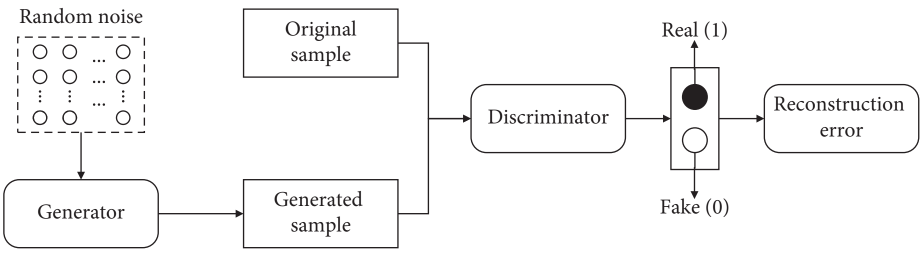


Figure 3: GAN's structure.

distribution to converge to a better result because the generator does not add any condition constraint, which causes a huge solution space when generating samples. Besides, as GAN is prone to input multiple random noise samples corresponding to the same type of sample generation in the process of sample generation, it is easy to reduce the diversity of generated samples and fall into model collapse (MC).

To complement each other's advantages, VAE's encoder and GAN's discriminator are combined to form a V-G network. Besides, when measuring the similarity, the original measurement of element error carried out by VAE is transformed into characteristic error measurement performed by GAN discriminator. For this, the V-G network can capture the data distribution characteristics easier. Therefore, using V-G for the training model not only can ensure that the diversity of sample generation is not restricted and improve its ability of mapping to original samples but also makes the discriminant result of similarity more precise. The V-G network structure is shown in Figure 4.

The V-G network in this paper is mainly used for network threat testing, and its application objects are mainly multisource heterogeneous network traffic data generated by the host, network, and server terminals. Due to the unique structural advantages of the V-G network, it can effectively extract data feature information during model training, so it can improve the accuracy of clustering and ensure higher accuracy of threat testing.

4. Network Security Threat Situation Assessment for IoT Based on the V-G Network

IoT applications and devices are vulnerable to various network threats because of the connection to the Internet. At present, common types of network threats include website information leakage, web attack threat, DDoS attack vulnerability, host commonly used service vulnerability, and system configuration security. Through the threat analysis of host and network traffic data, this paper aims to discover network threats and network vulnerabilities in time and carry out real-time network security situation threat assessment.

The network security threat situation assessment framework for IoT established in this paper is presented in Figure 5.

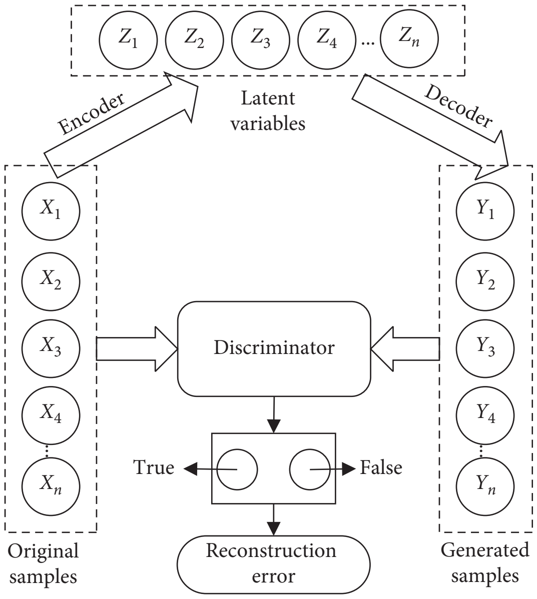


Figure 4: V-G's network structure.

The architecture includes five parts: assessment data set construction, data preprocessing, multisource data feature selection, network threat testing, and network threat situation assessment.

The steps of network threat quantitative assessment are as follows:

Step 1. Data acquisition: obtain the multisource network security traffic dataset as the evaluation data source.

Step 2. Data preprocessing: the original data is processed by the numerical method and feature specification to meet the requirements of model training and improve the utilization of the data.

Step 3. Feature selection: the characteristics of multisource network security traffic data are selected to reduce data redundancy.

Step 4. Threat testing: the unsupervised threat test model is used to test the threat and obtain the threat probability.

Step 5. Network threat situation assessment: obtain the threat severity and the threat impact according to the threat probability calculated in Step 4; then calculate the threat situation value and evaluate the overall situation of the network.

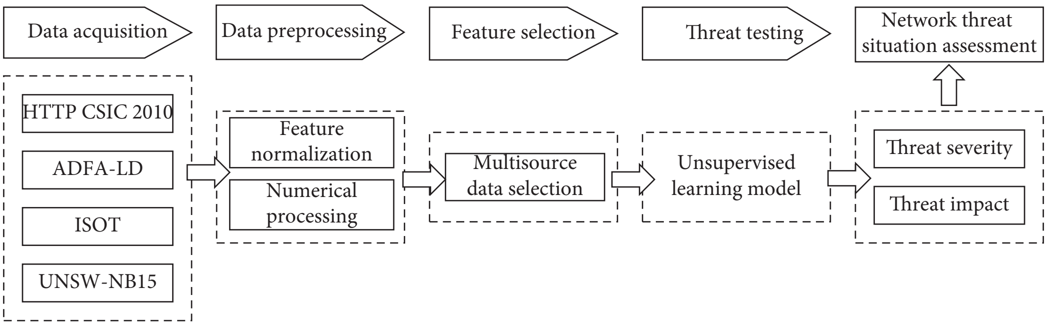


Figure 5: Network threat situation assessment framework.

4.1. Data Acquisition. IoT networks are susceptible to denial-of-service (DDoS) type of network attacks [16, 17]; in reality, however, IoT networks are facing various network attacks. To evaluate the network threat situation comprehensively, this paper selects four different types of network threat traffic datasets in the field of network security as the evaluation data sources; they are, respective, CSIC 2010 HTTP dataset based on web attack, ADFA-LD dataset based on Linux host exception, UNSW-NB15 dataset based on DDoS anonymous traffic attack, and ISOT dataset composed of mixed botnet traffic. Basic information on the four datasets is displayed in Table 1.

TP CSIC 2010 HTTP dataset is a set of normal and abnormal network attack traffic data automatically generated based on Web applications. It contains 36,000 normal requests and more than 25,000 exception requests. There are mainly three types of exception requests, which are divided into 16 attack categories.

ADFA-LD dataset is a network traffic dataset based on Linux host-level intrusion detection system, containing 5925 pieces of traffic data which are mainly divided into six attack categories: Hydra-FTP, Hydra-SSH, Adduser, Java-Meterpreter, Meterpreter, and Webshell.

ISOT dataset is composed of various botnet traffic and normal network data traffic which include 134916 pieces of traffic data divided into 19 characteristic categories: BytesAB, BytesBA, NpacketsAB, NpacketsBA, Duration, and so on.

UNSW-NB15 dataset is mainly composed of DDoS attacks in about an hour of anonymous traffic trace data; it contains 257673 traffic data, mainly divided into 9 types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.

Part of the network threat situation indicators contained in the four datasets is shown in Figure 6.

Figure 6 lists some threat situation indicators for V-G network testing. Besides, other types of threat indicators are not present in this paper, but they also are used for effective testing through the V-G network. The premise is to obtain data traffic sets that contain these threatened attacks because the model needs a lot of network traffic data as baseline data for model training.

Table 1: Basic information on four types of datasets.

Dataset	Data size	Category	Data type
HTTP CSIC 2010	61000	16	Web application
ADFA-LD	5925	6	Linux host exception
ISOT	134916	19	Hybrid botnet
UNSW-NB15	257673	10	DDoS anonymous attack

4.2. Data Preprocessing. Data preprocessing mainly includes two operations: numerical processing of character feature and feature normalization. It is necessary to carry out numerical processing for the symbolic data in the evaluation data source and convert all symbolic features into ordered numerical features since the training of the V-G network set requires digital feature vector as input. At the same time, to eliminate the dimension and facilitate the operations, all the numerical characteristics after the numerical treatment are normalized in the same interval.

4.2.1. Numerical Processing of Character Feature. Through the way of one-hot encoding, the 14 HTTP request feature classes of the CSIC 2010 HTTP dataset are transformed into numerical vectors. Specifically, transform 8 kinds of feature data, protocal, userAgent, accept, accept-Encoding, pragma, cacheControl, acceptCharset, and acceptLanguage, into numerical vectors of size between 0 and 1. Convert the 3 types of HTTP request data (GET, POST, and PUT) into binary eigenvectors (1, 0, 1), (1, 0, 0), and (1, 1, 0), respectively; moreover, the three types of URL extensions (JSP, GIF, and PNG) of the web application are converted into binary eigenvectors (1, 1, 1), (0, 1, 1), and (0, 1, 0), respectively; similarly, the 42-dimensional features of the UNSW-NB15 dataset are eventually converted into 196-dimensional binary numeric vectors after numeric processing.

4.2.2. Feature Normalization. There is a significant difference between the minimum and maximum values of some features while evaluating the data source. To suppress the negative impact of these outliers on the model training, the

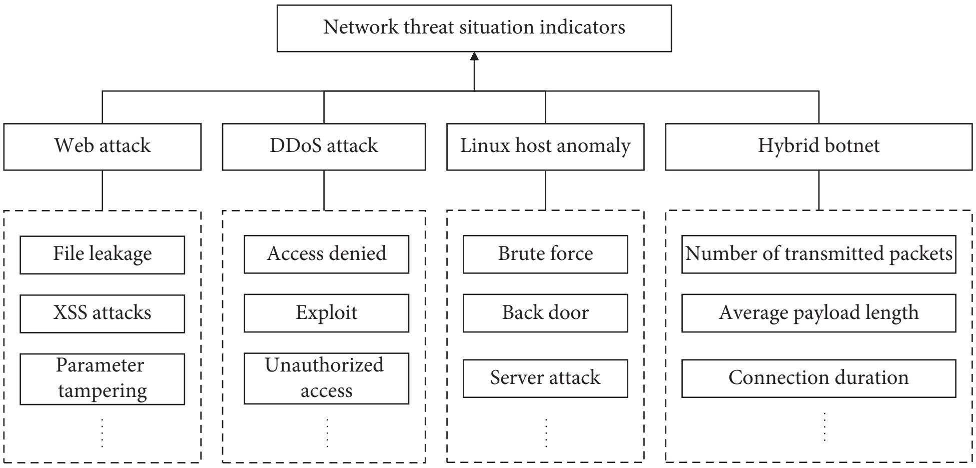


Figure 6: Part of the network threat situation indicators.

Max-Min scaling method is used to unify the feature values in the interval of [0, 1] and the formula is given as

$$x^{*} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \tag{1}$$

where $x*$ represents the normalized value of a certain class of features, x represents the initial eigenvalue, $x_{\min}$ is the minimum eigenvalue, and $x_{\max}$ is the maximum eigenvalue.

4.3. Multisource Dataset Feature Selection. To avoid the existing mass of redundant data of evaluating data source which may increase the overfitting risk of the V-G network in the training model and reduce the generalization ability of the model, this paper selects features of the evaluated data source which filter the unrelated features of the data source to ensure the high availability and the redundancy of data, improving the data clustering accuracy of all kinds of features in the V-G network and reducing the time complexity of model training.

In general, the feature selection process does not need to consider the structural characteristics of the data itself, but the flow data in the dataset used in this paper has the characteristics of clustering structure, so the three following factors should be considered before feature selection:

(1) V-G model training is a multifeature clustering process

(2) The data selected by features can keep the clustering structure characteristics of the flow data to the greatest extent

(3) The data selected by features can cover all possible clustering situations in a single dataset

From the above, the multicluster feature selection (MCFS) algorithm is selected for feature selection in this paper. MCFS is an unsupervised feature selection algorithm that does not rely on the data label information in the dataset. The feature selection process is divided into the five following steps.

Step 1. Constructing a k-nearest-neighbor graph. For each data point x_i corresponding to the graph with N vertices, a k-nearest-neighbor graph is constructed by searching for the k-nearest-neighbor points of x_i to obtain the local geometric structure features of the data distribution and the adjacency weight matrix W. In this paper, the Heat Kernel Weighting method is applied to calculate the adjacency weight matrix W among data points and the formula is as follows:

$$W_{ij} = e^{-\left(\left|x_i - x_j\right|^2/\sigma\right)}, \tag{2}$$

where x_i and x_j represent any two data points in the k-nearest-neighbor graph and σ is a fixed parameter.

Step 2. Spectral clustering embedded analysis. Define a diagonal matrix D whose diagonal elements are $D_{ij} = \sum_{j=1} W_{ij}$ and obtain the planar embedding structure of the data stream by calculating the generalized eigenvalue of Laplace matrix L:

$$\mathrm{LH}_k = \lambda \mathrm{Dh}_k, \tag{3}$$

where $L = D - W$ and $H = \{h_1, h_2, \ldots, h_k\}$ is the set of eigenvectors corresponding to the minimum generalized eigenvalues obtained through equation (3). Each column of H represents the planar embedding of any data point x_i and k represents the inner dimension of the data whose size is usually the number of clusters of the dataset.

Step 3. Sparse coefficient learning. After obtaining the planar embedding H of data points, to evaluate the importance of each feature in its corresponding data dimension (each column of H) and measure the ability of each feature to distinguish data clustering, MCFS takes the embedded h_k given by any column in H as a regression target and the objective function is represented by the following formula:

$$\min_{a_k} \left\| h_k - Q^T a_k \right\|^2 + \beta \left| a_k \right| \min_{a_k} \left\| h_k - Q^T a_k \right\|, \tag{4}$$

where a_k is an m-dimensional vector and Q is a matrix of $N \times M$. For minimizing the objective function, define the $L1$-norm of a_k as

$$\left|a_k\right| = \sum_{j=1}^{M} \left|a_{k,j}\right|, \tag{5}$$

where a_k includes the sparse coefficients used to approximate the different features of h_k. According to the penalty of $L1$-norm, the sparse coefficient of a_k will gradually shrink to zero when β is large enough. At this point, a subset of features that are most relevant to h_k will be selected.

Step 4. Calculate the MCFS score. Calculate k sparse coefficient vectors $\{a_1, a_2, ..., a_k\} \in \mathrm{R}^{\mathrm{M}}$ based on Step 3 for a dataset that contains k clusters, where each nonzero element a_k corresponds to d features. To select d effective features from k sparse coefficient vector, the MCFS score of each feature j is defined as

$$\mathrm{MCFS}(j) = \max_{k} \left|a_{k,j}\right|, \tag{6}$$

where $a_{k\ ,j}$ is the jth element of vector a_k.

Step 5. Feature selection. According to Step 4, calculate the MCFS scores of each class of features in the dataset and sort the MCFS scores of all features in a descending order and the first d important features will be selected.

4.4. Threat Testing. To detect the new attack threats that may appear in the network environment in real time, this paper applies a V-G network to perform network threat testing. The network threat situation test model built in this paper is shown in Figure 7.

The process of threat testing is mainly divided into four processing stages: network collection layer training, network parameters optimization, output layer reconstruction error training, and threat testing.

For the convenience of expression and analysis, let l represent a single V-G network layer, and let L_1 and L_2 represent the network collection layer and network output layer, respectively. L_1 is made out of ml and $L_1 = \{l_1, l_2, ..., l_m\}$. L_2 is a 3-layer variational autoencoder network with k input and output units. The detailed steps of the network threat testing process are designed as follows.

Step 1. Network collection layer training. Normal network traffic data is input to L_1 in batches for training after data preprocessing and multisource data feature selection. The training ends when it reaches a Nash equilibrium.

Step 2. Network parameters optimization. To overcome the parameters' tendency to fall into local optimization which is caused by the parameter tuning process with Gradient Descent (GD) method, Newton method (NM), Gauss Newton (GN) method, and other algorithms, this paper uses Levenberg-Marquardt (LM) optimization algorithm instead of GD and GN algorithm to carry out parameter tuning for the V-G network.

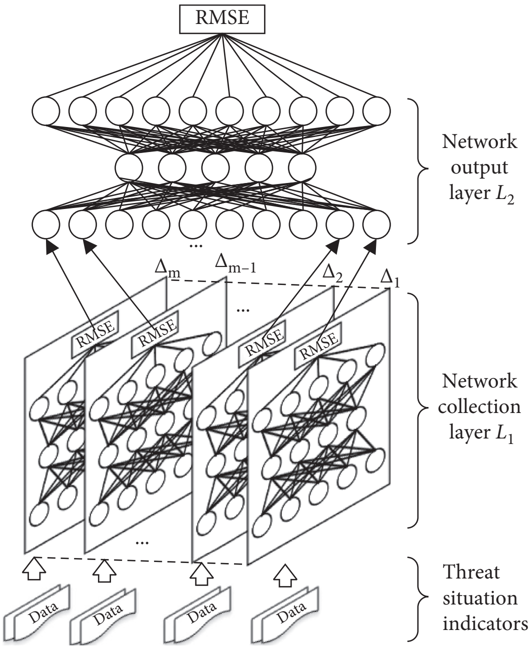


Figure 7: Network threat test based on the V-G network.

In the process of optimizing network parameters, four algorithms, GD, NM, GN, and LM, find the optimal function matching of high-dimensional data by minimizing the error sum of squares, namely, minimizing the objective function $f(x)$:

$$f(x) = \min \sum_{j=1}^{M} \sum_{i=1}^{N} f_{i,j}^{2}(x). \tag{7}$$

The gradient change of the objective function is

$$f'\left(x_{j,k}\right) = \sum_{j=1}^{M} \sum_{i=1}^{N} f_{i,j}(x) \frac{\partial f_{i,j}(x)}{\partial x_{j,k}}. \tag{8}$$

LM algorithm introduces the identity matrix I to avoid the irreversible phenomenon that may occur when the Jacobian matrix J (in GN algorithm) approximately represents the Hessian matrix H (in NM algorithm) and applies the damping factor μ to adjust the operation of the algorithm. LM algorithm combines GD algorithm and GN algorithm to dynamically tune parameters.

When optimizing the parameters, the optimization method is determined according to the gradient descent rate and the damping factor μ. If the gradient descent rate of the function is too slow, the damping factor μ increases. The GD algorithm is used to find the global optimal value:

$$x_{k+1}^{*} = x_k - (H + \mu I)^{-1} f'(x_k). \tag{9}$$

If the gradient descent rate of the function is too high, the damping factor μ decreases. The GN algorithm is used to find the global optimal value:

$$\begin{aligned} x_{k+1}^{*} &= x_k - (V + \mu I)^{-1} J^T f, \\ V &= J^T J. \end{aligned} \tag{10}$$

Step 3. Output layer reconstruction error training. The input item of the output layer network $\mathbf{L}_2$ comes from the 0-1 normalized reconstruction error value of the training output of each corresponding subnetwork in $\mathbf{L}_1$. The reconstructed error value of the output of $\mathbf{L}_1$ and $\mathbf{L}_2$ is calculated by the Root Mean Square Error (RMSE) function:

$$RMSE(\overrightarrow{x}, \overrightarrow{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{11}$$

where $\overrightarrow{x}$ and $\overrightarrow{y}$ represent the input sample vector and the generated sample vector, respectively, and n is the dimension of the input vector.

The training error set e^* output by L_1 can be expressed as $e^* = \{e_1, e_2, \ldots, e_m\}$. e^* will be the input item of L_2; then calculate training anomaly threshold η through the RMSE function when conducting error training.

Step 4. Threat testing. After the training of the V-G network collection layer and the training of output layer reconstruction error, the test dataset containing abnormal network traffic data is used for threat testing. Select m groups randomly in the same number of test samples v and take them as the input data of L_1. The test error output by L_1 in each test can be expressed as $\beta = \{\beta_1, \beta_2, \ldots, \beta_m\}$.

4.5. Network Threat Situation Quantitative Assessment. In this study, the quantitative assessment results of network threat situation are determined by two key factors that affect network security: threat severity and threat impact.

4.5.1. Threat Severity. In this paper, the unsupervised network model is used to analyze the characteristics of multisource network traffic data. After executing the threat tests, the normalized test error value β obtained according to the threat test results during each test is taken as the probability of threat occurrence:

$$\mathrm{TP}_i = \beta_i. \tag{12}$$

This paper refers to the "Overall Emergency Plans for National Sudden Public Incidents" [18] and develops the classification of network threat situations combined with the attack classification of the Snort Chinese user manual. The threat severity is divided into five levels in this paper: safety, low-risk, middle-risk, high-risk, and super-risk levels, corresponding to the five probability intervals of threat probability: 0.00~0.20, 0.21~0.40, 0.41~0.60, 0.61~0.80, and 0.81~1.00, respectively.

4.5.2. Threat Impact. To classify the degree of impact on the threat probability, the Common Vulnerability Scoring System (CVSS) [19, 20] is used to develop a classification table of threat impact (as shown in Table 2).

The formula for calculating the threat impact (TI) is defined as

$$\mathrm{TI}_i = \log_2\left(\frac{x_1 2^c + x_2 2^I + x_3 2^A}{3}\right). \tag{13}$$

C, I, and *A* represent the confidentiality, integrity, and availability of three threat impact indicators, respectively, and x_1, x_2, and x_3 represent the weight of quantified value of threat impact in three threat impact indicators, respectively.

Threat situation value (TSV, denoted as *T*) is determined by the threat probability and the threat impact. The calculation formula is as follows:

$$T = \frac{1}{n}\sum_{i=1}^{n}(\mathrm{TP}_i \times \mathrm{TI}_i). \tag{14}$$

5. Experiments and Results

The training and testing process based on the V-G network is carried out on the Ubuntu system, and the algorithm is implemented by Python programming language. The hardware environment of the experiment includes the Intel Core i7-7700 HQ processor, 8G RAM, and GTX 1050 graphics card, 16 GB.

5.1. Network Threat Test Results Analysis

5.1.1. Network Training. To prove the validity of the model in this paper, four networks, AE, VAE, GAN, and V-G, are, respectively, used to form a network set for model training. Four kinds of models use the same parameters for network training and the training data is the same set of normal network traffic data which ensures the comparability of the results. Model training is carried out when the number of layers of network collection is 5, 10, 15, 20, and 30.

The training anomaly threshold η output from four types of threat test models in the stage of model training under the different network layers is shown in Figure 8.

Figure 8 shows that, compared with the other three models, the V-G network obtains the minimum training error threshold η when the number of the network layers is 15, suggesting that refactoring capability for processing raw data of the V-G model is superior to the other three models.

In the process of model training, four optimization algorithms, GD, NM, GN, and LM, are used to optimize the model parameters of the V-G network, and the convergence of the optimization process of the four algorithms is shown in Table 3.

Table 2: Threat impact classification.

Threat impact	Probability interval	Impact indicators		
		Confidentiality (C)	Integrity (I)	Availability (A)
No-effect	0.00~0.40	0	0	0
Low-effect	0.41~0.80	0.22	0.22	0.22
High-effect	0.81~1.00	0.56	0.56	0.56

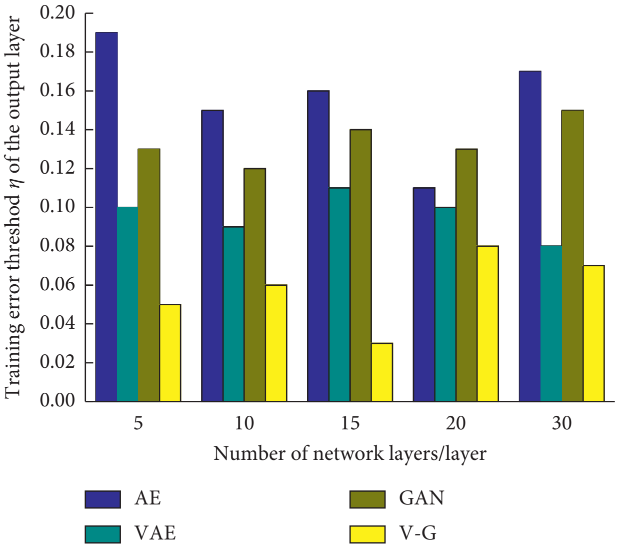


Figure 8: Four kinds of models training error threshold η.

Table 3: The convergence of different optimization algorithms.

Optimization algorithms	Iterations	Time (s)	RMSE
GD	220	350	0.35
NM	210	370	0.37
GN	200	320	0.32
LM	240	340	0.08

As can be seen from Table 3, compared with the other three algorithms, though the LM algorithm has more iterations and consumes more time, the Root Mean Square Error value is the smallest, indicating that the algorithm achieves a better convergence effect for the model which is more helpful for improving the accuracy of threat testing.

5.1.2. Network Testing. We conduct 200 groups of threat tests with random data of the same size, which is selected from the same test dataset. Four models, AE, VAE, GAN, and V-G, are used to carry out threat testing experiments, respectively. The normalized test error β obtained from the 10 groups of threat test experiments is shown in Figure 9.

As can be seen from Figure 9, compared with the other three types of models, the V-G network has the largest test error β when the number of network collection layers reaches 15 with the same test samples which indicate that its ability to detect network threats is more prominent.

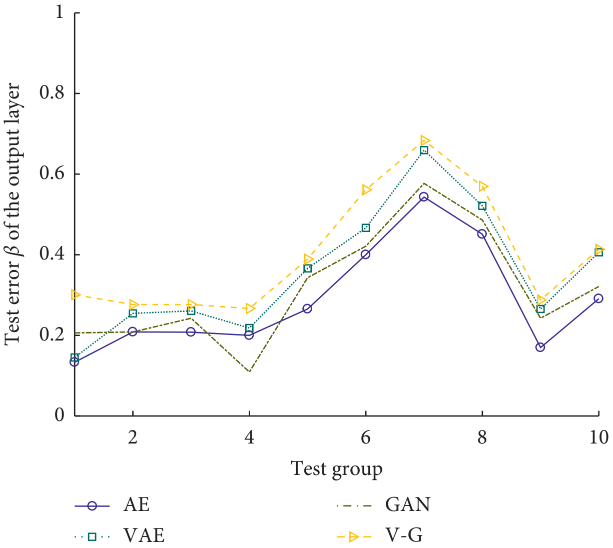


Figure 9: Threat test results of four kinds of models.

5.2. Network Threat Situation Quantitative Assessment Results Analysis. The test error β of each group is normalized to the interval of [0, 1] and is obtained through the process of network threat testing. The evaluation results of the threat severity and the threat impact of 10 groups of network threat situations are shown in Table 4.

Table 4: Evaluation results of the threat severity and the threat impact.

No.	Threat probability	Threat severity	Threat impact
1	0.187	Safety	No-effect
2	0.275	Low-risk	No-effect
3	0.238	Low-risk	No-effect
4	0.426	Middle-risk	Low-effect
5	0.262	Low-risk	No-effect
6	0.557	Mid-risk	Low-effect
7	0.685	High-risk	High-effect
8	0.504	Middle-risk	Low-effect
9	0.358	Low-risk	No-effect
10	0.281	Low-risk	No-effect

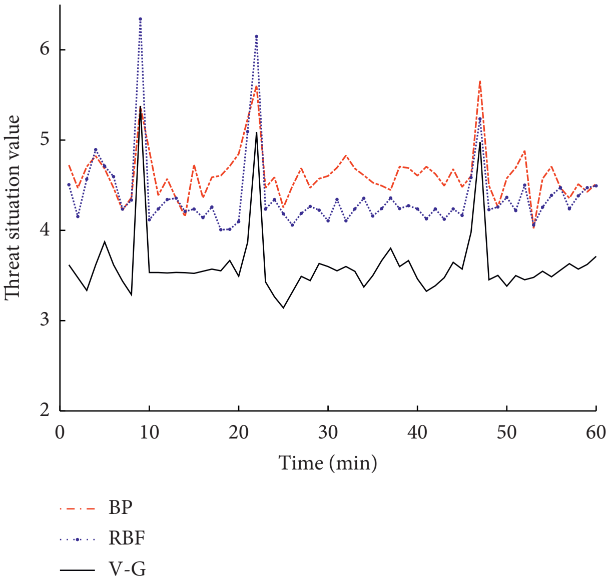


Figure 10: Comparison of threat situation values.

To increase the objectivity and authenticity of the evaluation results, the threat situation value was calculated, respectively, by Back Propagation (BP) [21] and Radial Basis Function (RBF) [22] methods and compared with the calculated results of the V-G network. The calculation results of the threat situation values of three types of methods in a certain period are displayed in Figure 10.

As can be seen from Figure 10, at 9 minutes, 22 minutes, and 47 minutes, the threat situation value shows a large range of changes, which indicates that the threat severity of the network is high at these moments and the network might be subjected to various types of attacks. It is found that, compared with the BP network and the RBF network in the three moments when the network is threatened, the method in this paper has a stronger capability of representing the features of network threats.

Besides, the curve of the V-G network is smoother than the other two networks, which indicates that the threat situation value calculated by the V-G network is more stable.

6. Conclusions

To overcome the limitations that traditional method of network threat situation assessment based on supervised learning needs to rely on data modeling label, this paper proposes a network threat situation assessment model based on unsupervised learning for IoT. This paper selects the multisource and heterogeneous datasets to simulate various network threats to IoT and calculates the threat situation value through quantifying the impact factors of network threat situation and then accomplishes the real-time situation of network threat assessment. The simulation experimental results show that the proposed method can evaluate the overall situation of network threats more intuitively and has a stronger characterization ability for network threats which can analyze the network security situation of IoT more precisely and take effective measures to reduce the risk of network threats. In the future, we will apply more network threat data that IoT will be confronted with on our proposed model, which will verify the general applicability of our proposed method.

Data Availability

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 work was supported by the Civil Aviation Joint Research Fund Project of the National Natural Science Foundation of China under Grant no. U1833107.

References

[1] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, “Internet of Things security: a survey,” *Journal of Network and Computer Applications*, vol. 88, pp. 10–28, 2017.

[2] Y. L. Mo and B. Sinopoli, “On the performance degradation of cyber-physical systems under stealthy integrity attacks,” *IEEE Transactions on Automatic Control*, vol. 64, no. 9, pp. 2618–2624, 2015.

[3] Z. L. Liu, B. Li, Y. Y. Huang et al., “New MCOS: Towards a practical multi-cloud oblivious storage scheme,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 4, pp. 714–727, 2019.

[4] Y. Huang, B. Li, Z. Liu et al., “ThinORAM: towards practical oblivious data access in fog computing environment,” *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 602–612, 2020.

[5] J. Li, Y. Huang, Y. Wei et al., “Searchable symmetric encryption with forward search privacy,” *IEEE Transactions on Dependable and Secure Computing*, vol. 99, p. 1, 2019.

[6] Z. Liu, J. Li, S. Lv et al., “EncodeORE: reducing leakage and preserving practicality in order-revealing encryption,” *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2020.

[7] Y. B. Leau, S. Manickam, and Y. W. Chong, “Network security situation assessment: a review and discussion,” in *Information Science and Applications*, pp. 407–414, Springer, Berlin, Germany, 2015.

[8] M. Yang, R. Jiang, T. L. Gao et al., “Research on cloud computing security risk assessment based on information entropy and Markov chain,” *International Journal of Network Security*, vol. 20, no. 4, pp. 664–673, 2018.

[9] H. Wang, Z. Chen, X. Feng et al., “Research on network security situation assessment and quantification method based on analytic hierarchy process,” *Wireless Personal Communications*, vol. 102, no. 2, pp. 1401–1420, 2018.

[10] H. Sallam, “Cyber security risk assessment using multi fuzzy inference system,” *International Journal of Engineering and Technology Innovation (IJETI)*, vol. 4, no. 8, pp. 13–19, 2015.

[11] Z. C. Wen, Z. G. Chen, and J. Tang, “Network security situation quantitative evaluation method based on information fusion,” *Journal of Beijing University of Aeronautics and Astronautics*, vol. 42, no. 8, pp. 1593–1602, 2016.

[12] W. Feng, Y. Q. Wu, and Y. X. Fan, “A new method for the prediction of network security situations based on recurrent neural network with gated recurrent unit,” *International Journal of Intelligent Computing and Cybernetics*, vol. 11, no. 4, pp. 511–525, 2018.

[13] F. N. He, Y. Q. Zhang, D. H. Liu et al., “Mixed wavelet-based neural network model for cyber security situation prediction using MODWT and Hurst exponent analysis,” in *Proceedings of the International Conference on Network and System Security*, pp. 99–111, Sapporo, Japan, December 2017.

[14] C. Doersch, “Tutorial on Variational Autoencoders,” 2016, http://arxiv.org/abs/1606.05908.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, pp. 2672–2680, Cornell University, New York, NY, USA, 2014.

[16] R. Vishwakarma and A. K. Jain, “A survey of DDoS attacking techniques and defence mechanisms in the IoT network,” *Telecommunication Systems*, vol. 73, no. 1, pp. 3–25, 2020.

[17] M. Roopak, G. Y. Tian, and J. Chambers, “Deep learning models for cyber security in IoT networks,” in *Proceedings of the 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0452–0457, Las Vegas, NV, USA, January 2019.

[18] The State Council of the People’s Republic of China, *Overall Emergency Plans for National Sudden Public Incidents*, The State Council of the People’s Republic of China, Beijing, China, 2006.

[19] P. Mell, K. Scarfone, and S. Romanosky, “Common vulnerability scoring system,” *IEEE Security and Privacy Magazine*, vol. 4, no. 6, pp. 85–89, 2006.

[20] Common Vulnerability Scoring System v3.0: Specification Document, 2020, https://www.first.org/cvss/specification-document.

[21] C. H. Tang and S. Z. Yu, “A network security situation prediction method based on likelihood BP,” *Computer Science*, vol. 36, no. 11, pp. 97–100, 2009.

[22] Z. Q. Lai, *Network Security Situation Prediction Model Based on Hybrid Optimization RBF Neural Network*, Lanzhou University, Lanzhou, China, 2017.