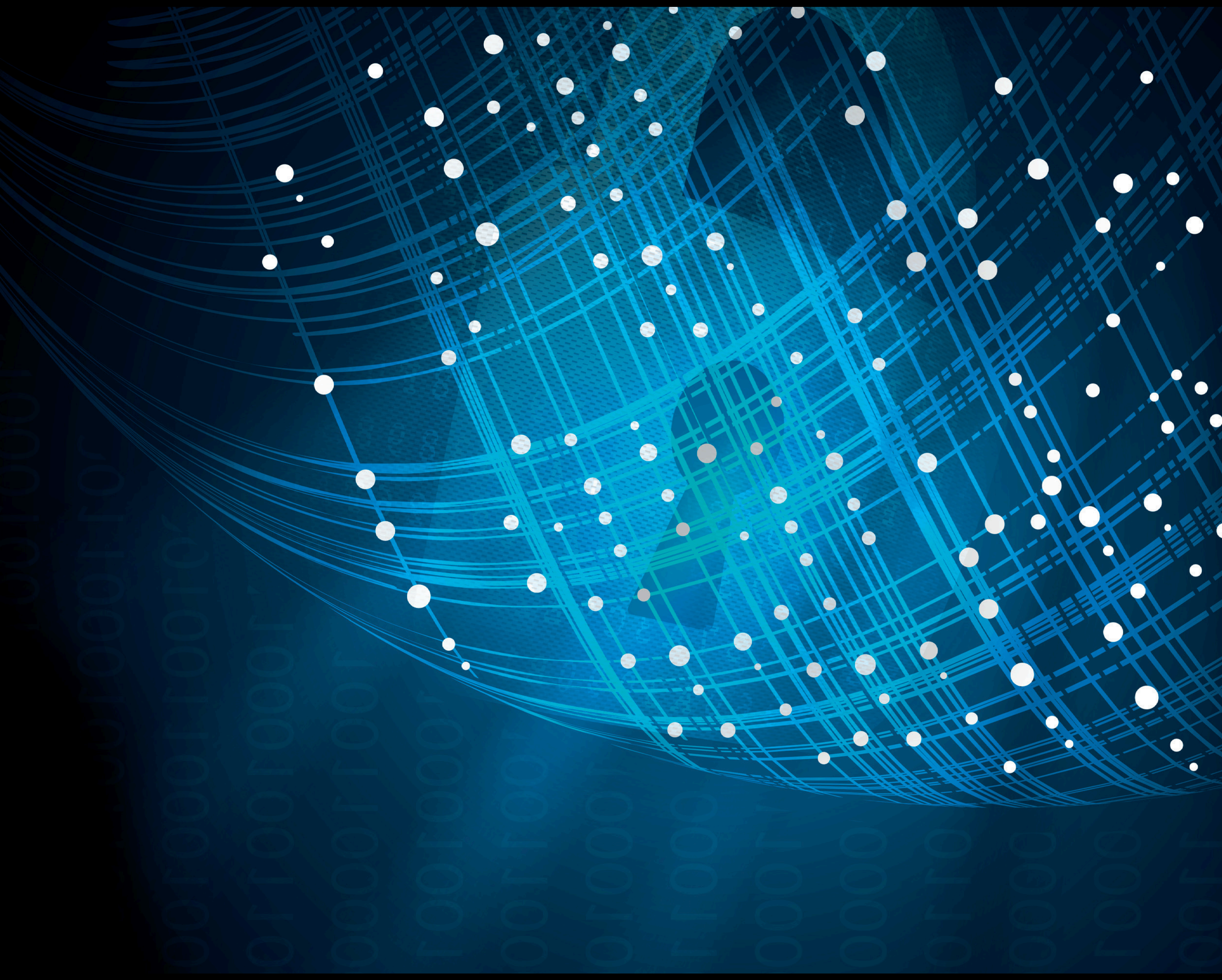
Security and Communication Networks

Identification of Attack Traffic Using Machine Learning in Smart IoT Networks

Lead Guest Editor: Muhammad Shafiq
Guest Editors: Shah Nazir and Xiangzhan Yu



Identification of Attack Traffic Using Machine Learning in Smart IoT Networks

Security and Communication Networks

Identification of Attack Traffic Using Machine Learning in Smart IoT Networks

Lead Guest Editor: Muhammad Shafiq
Guest Editors: Shah Nazir and Xiangzhan Yu

Contents

Hindawi
Security and Communication Networks
Volume 2022, Article ID 9804596, 4 pages
https://doi.org/10.1155/2022/9804596

Editorial

Identification of Attack Traffic Using Machine Learning in Smart IoT Networks

Muhammad Shafiq,[1] **Shah Nazir**,[2] **and Xiangzhan Yu**[3]

[1]*Guangzhou University, Guangzhou, China*
[2]*University of Swabi, Swabi, Pakistan*
[3]*Harbin Institute of Technology, Harbin, China*

Correspondence should be addressed to Muhammad Shafiq; srsshafiq@gmail.com

Received 18 March 2022; Accepted 18 March 2022; Published 11 April 2022

1. Introduction

Identifying attack traffic is very important for the security of Internet of Things (IoT) in smart cities by using machine learning (ML) algorithms. Recently, the IoT security research community has endeavoured to build anomaly, intrusion, and cyber-attack traffic identification models using machine learning algorithms for IoT security analysis. However, some critical and significant problems have not yet been studied in depth. One such problem is how to select an effective ML algorithm when there are a number of ML algorithms for a cyber-attack detection system for IoT security. Will early-stage traffic management give effective results if applied to IoT traffic management by using ML algorithms, or will this affect the performance of the ML model if several features are selected? Methods must avoid the risk of inaccuracy, inefficiency, and privacy leakage of machine learning techniques in IoT. The main objective of this Special Issue is to publish articles based on feature selection, algorithms, protocols, frameworks, and machine learning techniques in IoT that extend the current state of the art with innovative ideas and solutions in the broad area of security attack traffic detection and network traffic management. Theoretical and experimental studies for typical and newly emerging convergence technologies and cases enabled by recent advances are encouraged. High-quality review papers are also welcome.

1.1. Papers in This Special Section. A large number of papers were submitted to this Special Issue, and each paper was reviewed by three or more experts during the assessment process. After evaluating the overall scores, thirteen papers were selected for inclusion in this Special Issue.

Following is a brief description of the accepted papers:

(i) In paper [1], the Hybrid Monotone Empirical Mode Decomposition (HM-EMD) is a recent EMD-based method of generating intrinsic mode functions (IMFs) using the monotone property. The monotone property assumes that, at each IMF extraction step, local maxima and minima are either increasing or decreasing. Based on this property and along with the characteristics of EMD, the HM-EMD is a useful method for extracting hidden information in audio streams. This paper proposes an enhancement of HM-EMD based on the predicted correlation and periodicity between IMFs obtained from a modified intensity function. In addition, to prove its feasibility, they apply the method to detect short messages in music files. Experimental results show that, compared with traditional EMD and other recent EMD-based methods such as reduced iteration EMD, scalar-reduced iteration EMD, and modified iteration EMD, the proposed algorithm is superior to both nondominated sorting genetic algorithm II and fast nondominated sorting genetic algorithm II.

(ii) The sophisticated cyberattacks are evolving every day, and they are becoming difficult to be detected by conventional security measures. To defend the cyber-security of modern computer systems, researchers have been working on developing intelligent techniques to detect the cyberattacks. The AI techniques have been proved successful so far for many cybersecurity applications, such as intrusion detection, malware analysis, and attack forecasting. However, the complexity of these attacks grows rapidly and the AI techniques need to be continuously updated to detect these attacks. In this paper, the authors compare and analyze the approaches used in applying intelligent techniques in some applications of cybersecurity such as intrusion detection system (IDS), malware analysis, and network traffic monitoring. Based on the analysis, they define some open challenges in using AI for combating cybercrime. They also discuss the challenges and prospects by combing through over one hundred articles related to future research directions. Finally, they present their perspectives on how future research can improve the cyberattack detection system.

(iii) Paper [2] presents a communication cost optimization method based on security evaluation to address the problem of increased communication cost due to node security verification in the blockchain-based federated learning process. By studying the verification mechanism for useless or malicious nodes, they also introduce a double-layer aggregation model into the federated learning process by combining the competing voting verification methods and aggregation algorithms. The experimental comparisons verify that the proposed model effectively reduces the communication cost of the node security verification in the blockchain-based federated learning process.

(iv) In paper [3] entitled "Poor Coding Leads to DoS Attack and Security Issues in Web Applications for Sensors," researchers from the Department of Computer Engineering at Konkuk University, South Korea, identify common web programming errors that could lead to a denial-of-service (DoS) attack in web applications for sensors. The research team developed a testbed for two kinds of applications: one for single sensor data collection and the other for data retrieval from a sensor network. Their findings reveal how easily common coding blunders can expose critical infrastructure to unfortunate circumstances.

(v) Study [4] presents that in edge computing environments, a dynamic network failure happens frequently due to factors like time-varying nodes and service fluctuations. This failure often affects the performance of applications or even causes crashes. With the emergence of the model-based anomaly detection method, previous work has proven its effectiveness in helping edge computing systems to detect anomalous behaviors and recover from failures at runtime. However, these techniques often require ad hoc model regeneration for each new state of the system and are not suitable for unpredictable edge computing environments. To address this problem, they present AdaGUM—an adaptive graph updating model-based anomaly detection method. The proposed method uses a multidimensional graph to capture the interdependency between different elements of edge computing systems (e.g., software components) and then generates the subsequent state transition paths through random walks over graphs. The system behavior is then compared with the transition path based on behavior space. They evaluate AdaGUM with three real-world open-source systems (e.g., Spark Streaming) using real failures as anomalies and two criteria: accuracy and performance overhead that measures system resource consumption. The evaluation results show that AdaGUM can correctly detect 99% of anomalies with an average overhead of 3%.

(vi) The authors in the paper "TBSMR: A Trust-Based Secure Multipath Routing Protocol for Enhancing the QoS of the Mobile Ad Hoc Network" proposed a trust-based multipath routing protocol called TBSMR to enhance the MANET's overall performance. The main strength of the proposed protocol is that it considers multiple factors like congestion control, packet loss reduction, malicious node detection, and secure data transmission to intensify the MANET's QoS. The performance of the proposed protocol is analyzed through the simulation in NS2. The simulation results justify that the proposed routing protocol exhibits superior performance than the existing approaches.

(vii) The paper entitled "Compressed Wavelet Tensor Attention Capsule Network" proposes the compressed wavelet tensor attention capsule network (CWTACapsNet), which integrates multiscale wavelet decomposition, tensor attention blocks, and quantization techniques into the framework of capsule neural network. Specifically, the multilevel wavelet decomposition is in charge of extracting multiscale spectral features in the frequency domain; in addition, the tensor attention blocks explore the multidimensional dependencies of convolutional feature [5] channels, and the quantization techniques make the computational storage complexities be suitable for edge computing requirements. The proposed CWTACapsNet provides an efficient way to explore spatial-domain features, frequency-domain features [6], and their dependencies which are useful for most texture classification tasks. Furthermore, CWTACapsNet benefits from quantization techniques and is suitable for edge computing applications. Experimental results on several texture datasets show that the proposed CWTACapsNet outperforms the state-of-the-art texture classification methods not only in accuracy but also in robustness.

(viii) In the paper entitled "Employing Deep Learning and Time Series Analysis to Tackle the Accuracy and Robustness of the Forecasting Problem," the authors apply time series to predict the crime rate to facilitate practical crime prevention solutions. Machine learning [7, 8] can play an important role in better understanding and analysis of the future trend of violations. Different time-series forecasting models have been used to predict the crime. These forecasting models are trained to predict future violent crimes. The proposed approach outperforms other forecasting techniques for daily and monthly forecast.

(ix) In the paper entitled "Cuckoo Search-based SVM (CS-SVM) Model for Real-Time Indoor Position Estimation in IoT Networks," the authors proposed a hybrid technique using Cuckoo Search-based Support Vector Machine (CS-SVM) for real-time position estimation. Cuckoo search is a nature-inspired optimization algorithm, which solves the problem of slow convergence rate and local minima of other similar algorithms. The Wi-Fi [9] RSSI fingerprint dataset of the UCI repository having seven classes is used for simulation purposes. The dataset is preprocessed by min-max normalization to increase accuracy and reduce computational speed. The proposed model is simulated using MATLAB and evaluated in terms of accuracy, precision, and recall with K-nearest neighbor (KNN) and support vector machine (SVM). Moreover, the simulation results show that the proposed model achieves a high accuracy of 99.87%.

(x) Although access control is one of the most important and effective methods for time-series data security, most existing access control models focus on the function of holding or managing data. However, the method of controlling transmission path is ignored. To maximize data security, the authors proposed an IoT time-series data security model based on thermometer encoding and proposed a new hyper-chaotic system as the source-generating system to build an adversarial attack model by using input parameter sensitivities detection. The authors designed a new adversarial attack model which can prevent input parameter sensitivity detection for realizing the maximum data security in the transmission process.

(xi) Due to the development of the digital economy, Internet of Things (IoT) has been widely used in various fields. The data security of IoT has become a hot research topic. Generally, the data security of IoT cannot be guaranteed without encryption. Time series encryption can better protect IoT data, but it is still a challenge for time-series encryption, especially in the case that there is an adversary attack. Therefore, the authors design an adversarial attack model and then propose an IoT time-series data security model based on thermometer encoding. Finally, the authors evaluate the performance of the proposed model through experiments and compare it with other encryption algorithms.

(xii) The last paper proposes an anomaly detection [10] algorithm selection service (ADS) with genetic algorithm (GA) and tsfresh tool. For IoT stream data, it requires that the anomaly detection algorithm can provide good recommendation for easy operation for IoT devices in factory automation systems. Moreover, the proposed method can compare suitable detection models from 28 candidates that are introduced by tsfresh tool with suitable input parameters which are determined by GA methods. The experiments are conducted and ADS system has achieved good results for anomaly detection which can be a good reference for other researchers or users for their solutions.

Conflicts of Interest

The editors declare that they have no conflicts of interest.

Acknowledgments

The editors would like to express their sincere thanks to all the authors who made this Special Issue possible and to the reviewers for their thoughtful contributions.

Muhammad Shafiq
Shah Nazir
Xiangzhan Yu

References

[1] J. Lou, Z. Xu, D. Zuo, and H. Liu, "Feature Extraction Method for Hidden Information in Audio Streams Based on HM-EMD," *Security and Communication Networks*, vol. 2021, Article ID 5566347, 12 pages, 2021.

[2] S. Xuan, M. Jin, X. Li, Z. Yao, W. Yang, and D. Man, "DAM-SE: A Blockchain-Based Optimized Solution for the Counterattacks in the Internet of Federated Learning Systems," *Security and Communication Networks*, vol. 2021, Article ID 9965157, 14 pages, 2021.

[3] K. B. Jalbani, M. Yousaf, M. S. Sarfraz, R. Jamili Oskouei, A. Hussain, and Z. Memon, "Poor Coding Leads to DoS Attack and Security Issues in Web Applications for Sensors," *Security and Communication Networks*, vol. 2021, Article ID 5523806, 11 pages, 2021.

[4] X. Yu, C. Shan, J. Bian, X. Yang, Y. Chen, and H. Song, "AdaGUM: An Adaptive Graph Updating Model-Based Anomaly Detection Method for Edge Computing Environment," *Security and Communication Networks*, vol. 2021, Article ID 9954951, 12 pages, 2021.

[5] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[6] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2020.

[7] A. Krysovatyy, H. Lipyanina-Goncharenko, S. Sachenko, and O. Desyatnyuk, "Economic crime detection using support vector machine classification," *CEUR Workshop Proceedings*, vol. 2917, pp. 830–840, 2021.

[8] N. Usman, S. Usman, F. Khan et al., "Intelligent dynamic malware detection using machine learning in IP reputation for forensics data analytics," *Future Generation Computer Systems*, vol. 118, pp. 124–141, 2021.

[9] M. F. Khan, G. Wang, and M. Z. A. Bhuiyan, "Wi-Fi frequency selection concept for effective coverage in collapsed structures," *Future Generation Computer Systems*, vol. 97, pp. 409–424, 2019.

[10] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5566347, 12 pages
https://doi.org/10.1155/2021/5566347

Research Article

Feature Extraction Method for Hidden Information in Audio Streams Based on HM-EMD

Jiu Lou, Zhongliang Xu, Decheng Zuo, and Hongwei Liu

Harbin Institute of Technology, School of Computer Science and Technology, Harbin 150001, China

Correspondence should be addressed to Decheng Zuo; zuodc@hit.edu.cn

Received 14 February 2021; Revised 22 March 2021; Accepted 21 July 2021; Published 2 August 2021

Academic Editor: Shehzad Ashraf Chaudhry

Using fake audio to spoof the audio devices in the Internet of Things has become an important problem in modern network security. Aiming at the problem of lack of robust features in fake audio detection, an audio streams' hidden feature extraction method based on a heuristic mask for empirical mode decomposition (HM-EMD) is proposed in this paper. First, using HM-EMD, each signal is decomposed into several monotonic intrinsic mode functions (IMFs). Then, on the basis of IMFs, basic features and hidden information features HCFs of audio streams are constructed, respectively. Finally, a machine learning method is used to classify audio streams based on these features. The experimental results show that hidden information features of audio streams based on HM-EMD can effectively supplement the nonlinear and nonstationary information that traditional features such as mel cepstrum features cannot express and can better realize the representation of hidden acoustic events, which provide a new research idea for fake audio detection.

1. Introduction

With the development of the Internet of Things (IoT) technology, an increasing number of audio and video acquisition devices are now connected to the Internet. Fake audio becomes an increased new threat on voice interfaces due to the recent breakthroughs in speech synthesis and voice conversion technologies. Therefore, the detection of fake audio has become a new hot issue of network security [1, 2]. There are mainly two methods of audio forgery. One is to generate spoofed utterances using text-to-speech (TTS) and voice conversion (VC) algorithms, which is also called logic access (LA) [3] and the other is the use of professional replay devices to get spoof attack, which is also known as physical access (PA) [4]. There are more diversity of audio forgery means and more difficulty in fake audio detection [5]. In this paper, an audio streams hidden feature extraction method based on HM-EMD is proposed and used to extract features of audio streams to detect fake audio.

At present, the fake audio detection is mainly based on acoustic features to build classification model. Linear frequency cepstral coefficients (LFCC) [6], constant-Q cepstral coefficients (CQCC) [7], and mel frequency cepstral coefficients (MFCC) [8] are commonly used in fake audio detection. However, these features are based on fixed filter banks; none of these acoustic features are able to generalize well on unknown spoofing technologies. Subsequently, the end-to-end deep learning method to detect the fake audio is gradually concerned by researchers. Alejandro et al. proposed a cyclic neural network based on optical convolution gate to extract the shallow features at the frame level and the deep features of sequence dependence at the same time [9]. Zeinali et al. used VGg and light CNN to detect fake audio [10]. However, this end-to-end deep learning approach requires large, evenly distributed datasets and relies on certain types of fake audio.

Through the analysis of forged audio, it is found that the AI-based fake audio technology focuses more on speech content and ignores the background sound in an audio stream [11]. The background sound will also change during the replay spoofing. Therefore, the construction of features representing the hidden information in audio scenes can be used to detect the fake audio [12].

In order to focus on local level details of the signal in terms of specific regions (which may be highly discriminative), empirical-mode-decomposition- (EMD-) based approach is explored. EMD has superior time-frequency resolution performance in nonlinear unsteady signal processing and has been applied in counterfeit audio detection [13].

However, the traditional EMD method has a few disadvantages, including mode aliasing and the inconsistency of IMF dimensions after signal decomposition. Hence, accurately estimating the IMF range of a certain frequency distribution is difficult. In 2005, Deering and Kaiser proposed the ensemble empirical mode decomposition (EEMD) decision method [14], which attempts to solve the problem of mode aliasing by introducing Gaussian white noise into the signal to be decomposed. In EEMD, the attributes of Gaussian white noise should be adjusted artificially. However, the Gaussian white noise leaves traces in the IMF decomposed from the signal, thereby resulting in low signal restoration accuracy and extensive calculations. Time-varying filtering-based empirical mode decomposition (TVF-EMD) uses the b-spline time-varying filter for mode selection and thus solves the problem of mode aliasing to a certain extent. However, TVF-EMD must calculate the cutoff frequency first, thus leaving the problem of dimension inconsistency unsolved [15].

To sum up, in order to make full use of the time-frequency analysis advantages of EMD, it is necessary to solve the modal aliasing and frequency inconsistency problems existing in EMD itself. In this paper, a heuristic empirical mode decomposition (HM-EMD) method is proposed to improve the purity of IMFS and solve the problem of inconsistency between mode mixing and IMF dimension. Then, the acoustic hidden component features (AHCF) of the audio stream were constructed and used to locate the acoustic events in audio stream in the acoustic stream classification dataset A of DCASE [16]. Fake audio detection is implemented on ASVSpoof2019 dataset [17]. The experimental results show that the basic features and AHCFs of the audio streams based on HM-EMD can represent the audio background which help to verify the types of the fake audio.

The paper consists of five parts. The first part is an introduction of audio streams. The second part mainly introduces the principle of HM-EMD. The third part describes the mining of hidden information in audio streams based on the proposed HM-EMD. The fourth part presents the results of classification of audio streams on the basis of HM-EMD. The fifth part summarizes the characteristics of the proposed method and presents future research directions.

2. Heuristic Mask for Empirical Mode Decomposition (HM-EMD)

In this section, the classical empirical mode decomposition (EMD) method is first introduced and then follows the analysis of mode aliasing in EMD; finally, the solution of mode aliasing based on heuristic mask signal is proposed in detail.

2.1. Empirical Mode Decomposition Method

2.1.1. Empirical Mode Decomposition. EMD can decompose the original signal $x(t)$ $(t \in N, N = \{0, 1, \ldots n\})$ into a series of IMFs whose upper and lower envelopes have a mean value of 0. This decomposition method does not need to preset basis functions (such as Fourier transform or wavelet analysis), but the IMFs should satisfy the following formulas:

$$\left|\mathrm{Num}_{\mathrm{extream}} - \mathrm{Num}_{\mathrm{cross}}\right| \le 1, \quad (1)$$

$$\sum_{t\in N} B_{\max}(t) + \sum_{t\in N} B_{\min}(t) = 0, \quad (2)$$

where $\mathrm{Num}_{\mathrm{extream}}$ is the number of extreme points of the data sequence and $\mathrm{Num}_{\mathrm{cross}}$ is the number of zero crossings; $B_{\max}(t)$, $B_{\min}(t)$ are the upper and lower envelopes by cubic spline interpolation with the maximum and minimum points as the control points, respectively. Formula (1) represents the narrow-band constraint condition of the IMF, and formula (2) represents the local symmetry constraint condition. The process of EMD decomposition to obtain an IMF can be expressed as follows (Algorithm 1).

2.1.2. Modal Aliasing of EMD. However, the most significant drawback of EMD is modal aliasing, as shown in Figure 1. Figure 1(b) shows the FFT spectrum corresponding to each IMF in Figure 1(a). It can be seen from the figure that each FFT spectrum contains multiple signals of different frequencies, which means a single IMF contains signals of different frequencies or signals of the same frequency that appear in different IMF components. These are modal aliasing. The main reason for modal aliasing is the absence of extreme value or the inconsistent distribution of extreme value, which makes the variation trend error between the spectral envelope obtained by interpolation and the real signal is too large. At this time, the time-domain signal does not meet the narrow-band requirements of IMF decomposition, resulting in mode aliasing.

In order to solve this problem, mask signal $s(t)$ is usually created to compensate the missing extreme value, and then the values are given, respectively:

$$x_{+}(t) = xt + st, \quad (3)$$

$$x_{-}(t) = xt - st. \quad (4)$$

For $x_{-}(t)$ and $x_{+}(t)$, EMD is performed to obtain the natural mode functions $r_{\mathrm{IMF}-}(t)$, respectively. The final IMF is defined as follows:

$$r_{\mathrm{IMF}}(t) = \frac{r_{\mathrm{IMF}+}(t) + r_{\mathrm{IMF}-}(t)}{2}. \quad (5)$$

It can be seen from the above that the extreme value distribution of mask signal $s(t)$ is very important to solve the modal aliasing problem. White noise is usually used as mask signal $s(t)$, but this method does not make full use of the properties of the signal itself and cannot adapt to a variety of signal contents. Therefore, this paper proposes a heuristic mask for empirical mode decomposition method. This

```
Input: original signal $x(t)$, supposed IMF number $i$
Output: intrinsic mode functions, IMF
(1) $i = 1$, $x^1(t) = x(t)$.
(2) Get the extremum points $\{u_1^{\max}, u_1^{\min}, u_2^{\max}, \ldots\}$ of signal $x^i(t)$, calculate the upper and lower envelope $B_{\max}(t)$, $B_{\min}(t)$ by cubic spline interpolation with the maximum and minimum points as control points, and get the average value of upper and lower envelope $B_{\text{mean}}(t)$ at every points.
(3) $r(t) = x^i(t) - B_{\text{mean}}(t)$. If $r(t)$ satisfies formulas (1) and (2), then $r(t)$ is taken as the $i$th IMF signal $r_{\text{IMF}}^i(t)$, $i = i + 1$; if not, repeat step 2 and 3 for signal $r(t)$.
(4) $x^i(t) = x^{i-1}(t) - r_{\text{IMF}}^{i-1}(t)$. Return to step 1 until the termination condition is satisfied.
```

Algorithm 1: Empirical mode decomposition.

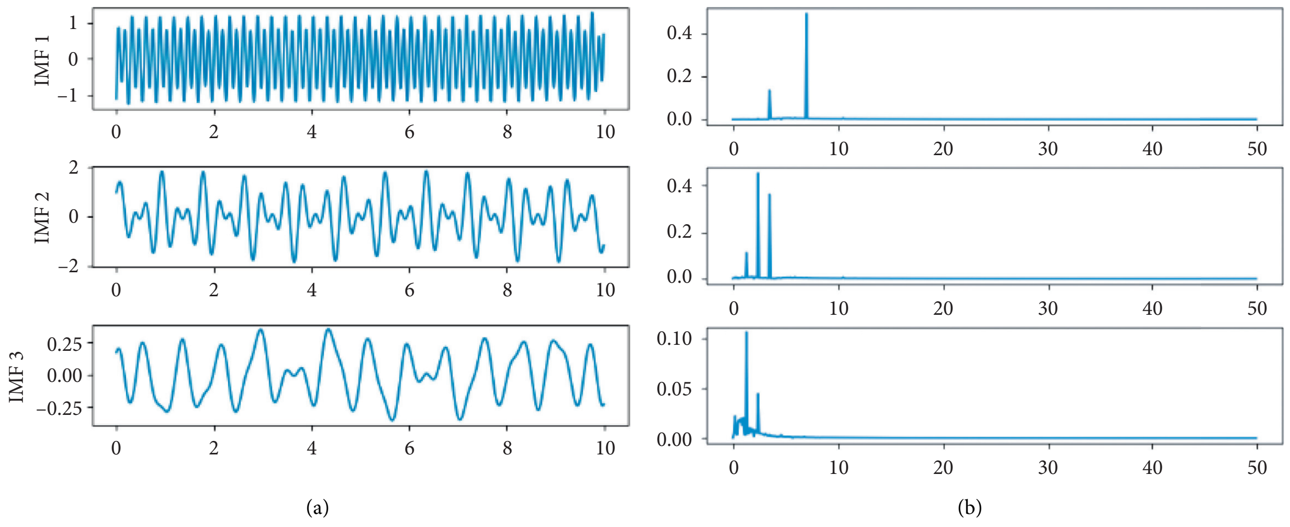


Figure 1: EMD results for signal $x_1(t) = \sin 2\pi * 2.4t + \sin 2\pi * 3.5t + \sin 2\pi * 7t$: (a) IMFs; (b) FFT spectrum.

method makes full use of the structural attributes of the signal itself to construct variable analysis window and mask signals. The specific principle and implementation process are as follows.

2.2. Heuristic Mask Signals

2.2.1. Basic Principle Analysis. The signal properties need to be established prior to EMD. A time-varying FM/AM model can be used to express any nonstationary signal; that is,

$$x(t) = At \sin(\omega(t)), \tag{6}$$

where $a(T)$ is the envelope function and $\omega(T)$ is the phase function. The analytical signal is

$$z(t) = xt + jH[x(t)]. \tag{7}$$

Here, $H[\cdot]$ denotes the Hilbert transform. We calculate the instantaneous phase $\omega(t) = \arctan(H[x(t)]/x(t))$ and instantaneous frequency $f_{IF}t = (1/2\pi)(\mathrm{d}[\omega(t)]/\mathrm{d}t)$. Using Hilbert transform, we can separate the AM and FM components of the IMF to achieve the purpose of modal separation.

For the single component mode, the instantaneous frequency $f_{IF}t$ should be nearly linear, while the variation range of $\omega(t)$ should be considerably small. When mode aliasing occurs, $f_{IF}t$ should clearly change without consideration of the end points. Especially, for hidden components, a jump of $f_{IF}t$ occurs at the time point of concealment. We constructed a variable analysis window according to the time-frequency characteristics of instantaneous frequency. Then, we divided the signal into several parts.

If $f_{IF}t$ of the segmented signal is still unstable; then, the modal separation problem can be transformed into the $(\mathrm{d}[\omega(t)]/\mathrm{d}t)$ minimisation problem, in which the bandwidth of $\sin(\omega(t))$ is minimised. The bandwidth calculation method for nonstationary signals can be obtained by the Carson rule:

$$\mathrm{BW}_{\text{AM-FM}} = 2\Delta f + f_{\text{FM}} + f_{\text{AM}}, \tag{8}$$

where Δf is the deviation of the instantaneous frequency from its mean value and f_{AM} and f_{FM} denote the frequencies of the AM and FM signals, respectively. We can make $\Delta f = 0$ to minimise the bandwidth. In other words, the decomposition frequency of each IMF is expected to be equal to the centre frequency of the instantaneous frequency, that is, equal to the mean value of the instantaneous frequency $\overline{f_{IF}t}$. Then, a mask signal with the same frequency as $\overline{f_{IF}t}$ can be selected and the number of IMFs required can be determined.

2.2.2. Algorithm Description. The HM-EMD algorithm comprises the following steps: variable analysis window construction and mask signal construction.

(1) Variable Analysis Window Construction. The jump point t_i should be picked such that formula (9) is satisfied:

$$\left|f_{IF}t_i - f_{IF}t_{i+1}\right| + \left|f_{IF}t_{i-1} - f_{IF}t_i\right| > \mu_{\Delta f_{IF}t} + \rho\varepsilon_{\Delta f_{IF}t}, \quad (9)$$

where $\Delta f_{IF}t$ is the difference in instantaneous frequencies at t_i, $\mu_{\Delta f_{IF}t}$ is the mean value of $\Delta f_{IF}t$ at all time points, $\varepsilon_{\Delta f_{IF}t}$ is the variance, and ρ is the variable parameter. The original signal is divided into two parts by the time division points t_i and decomposed by EMD independently.

(2) Mask Signal Construction. The sine signal is a common form of a mask signal, and its amplitude and frequency should be determined. As analysed in Section 2.1, the frequency is determined as the average instantaneous frequency $\overline{f_{IF}}$. Hence, the amplitude is also determined as the mean value $\overline{A_{IF}}$ of the instantaneous amplitude. Then, the mask signal s t is defined as

$$st = \overline{A_{IF}} \sin 2\,\pi\overline{f_{IF}}t, \quad (10)$$

where $\overline{A_{IF}} = (1/n)\sum_{t=1}^{n}\sqrt{r_{IF}(t)^2 + H(r_{IF}(t)^2)}$ and $\overline{f_{IF}} = (1/n)\sum_{t=1}^{n}(\mathrm{d}/\mathrm{d}k)(\arctan(H(r_{IF}(t))/r_{IF}(t)))$.

Then, IMFs can be refreshed by formulas (3)–(5), in which the number of IMFs is determined by $\overline{f_{IF}}$ and f_c is the sampling frequency. The algorithm flow is as follows (Algorithm 2):

3. HM-EMD-Based Acoustic Scene Classification

The audio stream contains the hidden acoustic events that can represent the acoustic scene. In this section, HM-EMD is first used to decompose the acoustic scene signals, and the IMF of hidden acoustic events in these acoustic scene signals is analysed. According to the analysis results, a full-band IMF hidden component feature is proposed to represent the hidden acoustic events. Finally, the process of acoustic scene classification using these features is given in detail.

3.1. Acoustic Scene Signal Analysis by HM-EMD. When processing the original signal with HM-EMD, the variable analysis window and mask signal are used to intervene the decomposition of the original signal. The frame length is selected according to the frequency structure of the signal itself, while the frequency domain components corresponding to each IMF are relatively independent, which provides higher interpretability of the features. The instantaneous frequency and amplitude of each IMF also contain all information of IMF components, which means that the instantaneous frequency and amplitude of all IMF components contain most of the information of the signal to be analysed and can be directly used as the basic characteristics of the signal. Figure 2 shows the time-domain waveforms of some typical IMFs with hiding acoustic events in the ambient audio stream, in which only the most significant one of all IMF waveforms is shown. It can be seen that the time-domain waveform characteristics of these events are very obvious, the extreme value and over-average rate are very different, and they are distributed in low, medium, and high frequency bands. Therefore, this paper proposes a full-band IMF hiding component features, which can distinguish them well, to effectively improve the effect of ambient audio stream recognition algorithm. The feature calculation method is shown in Section 3.2.

3.2. Mutagenic Component Features. Figure 2 shows various hidden components in the acoustic scene data. On the one hand, the hidden components cause a significant interference to the signal spectrum, thereby greatly affecting the ambient audio stream recognition effect based on traditional spectrum features (such as MFCC). On the other hand, the types and characteristics of hidden components corresponding to different ambient audio streams also exhibit significant differences. These hidden components are closely related to the types of acoustic events. The features constructed on the basis of hidden components can help to distinguish ambient audio streams. For a hidden component, its frequency, amplitude, and change mode information can effectively reflect its essential attributes. Almost all of such information can be reflected by the envelope shape of the IMF obtained by decomposition. Therefore, we design a set of HACFs. Based on the IMF decomposed by HM-EMD, the features extract the relevant information of hidden components, including the shock intensity feature SH and over-average feature average crossing rate (ACR).

3.2.1. Shock Intensity Feature (SH).

$$\begin{aligned}\mathrm{SH}_{\max j} &= \max\left(r_{\mathrm{IMF}j}^{\mathrm{up}}(t)\right),\\ \mathrm{SH}_{\min j} &= \min\left(r_{\mathrm{IMF}j}^{\mathrm{up}}(t)\right),\end{aligned} \quad (11)$$

where max $(r_{\mathrm{IMF}j}^{\mathrm{up}}(t))$ is the upper limit of the signal amplitude in the jth IMF and min $(r_{\mathrm{IMF}j}^{\mathrm{up}}(t))$ is the lower limit. Both limits represent the change intensities of the hidden components relative to the steady components for measuring the changes in signal amplitude. As the sum of the mean values of the upper and lower envelopes of the IMF is 0, the signal is symmetrical along the time axis, and the information carried by the upper and lower envelopes is almost the same. Therefore, a one-sided envelope is enough to ensure the consistency of the symbols of the two values. The superscript means that the upper envelope is used for calculation.

```
    Input: signal $x(t)$, supposed IMF number $i$
    Output: intrinsic mode function, IMF
(1) $x_1(t) = x(t)$, $i = 1$.
(2) Get the first IMF of the signal residual $x_i(t)$, calculate the mean and variance of $\Delta f_{IF} t$, and use formula (8) to determine whether
    there is a hiding jump point. Variable analysis window is constructed according to the hiding jump point and $x_i(t)$ is segmented.
(3) Construct mask signal for each IMF$_i$: $s_i t = \overline{A_{IF_i}} \sin 2\pi \overline{f_{IF_i}} t$.
(4) Do EMD on $x_{i+}t = x_i(t) + s_i t$ and $x_{i-}t = x_i(t) - s_i t$; get the first IMF $r_{\mathrm{IMF}_i+}(t)$ and $r_{\mathrm{IMF}_i-}(t)$.
(5) Let $r_{\mathrm{IMF}_i}(t) = (r_{\mathrm{IMF}_i+}(t) + r_{\mathrm{IMF}_i-}(t))/2$, and splice all the divided pieces.
(6) $i = i + 1$, $x_i(t) = x_{i-1}(t) - r_{\mathrm{IMF}_i}(t)$, return to step 2, until $\overline{f_{IF_i} t} < (f_c/2i)$, or no new IMF is required.
```

Algorithm 2: Heuristic empirical mode decomposition with a masking signal.

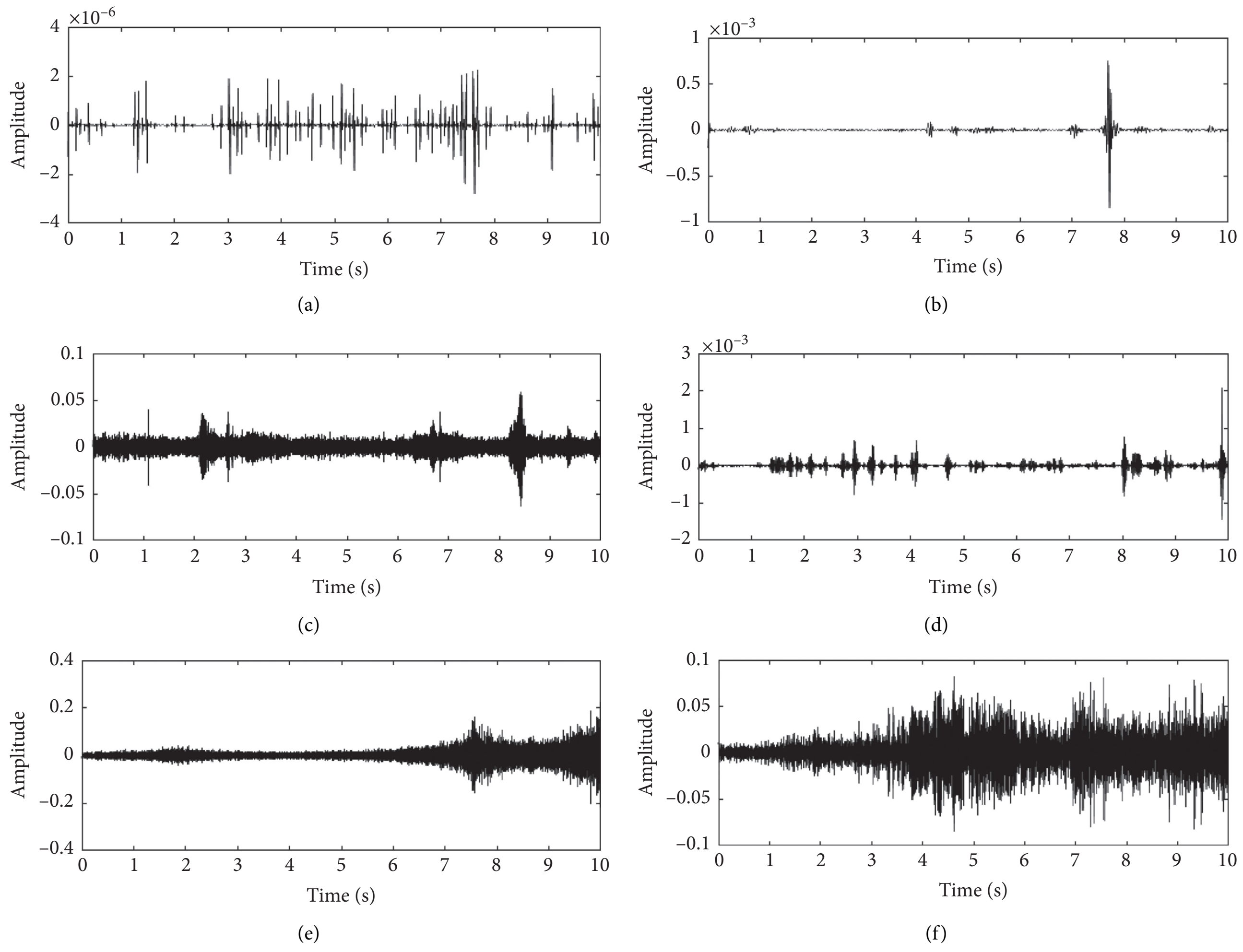


Figure 2: IMF waveforms with significant hidden components in different environments of the audio stream. (a) Airport luggage roller: IMF16, low freq. (b) Metro rail joint collision: IMF14, low freq. (c) Chirm: IMF1, high freq. (d) Steps: IMF10, medium freq. (e) Vehicles from far to near: IMF1, high freq. (f) Tram acceleration: IMF4, medium and high freq.

3.2.2. ACR Feature.

$$\mathrm{ACR}_i = \frac{1}{2T} \sum_{i=2}^{T} \left| \mathrm{sgn}\left[r^{\mathrm{up}}_{\mathrm{IMF}j}(t) - \overline{r^{\mathrm{up}}_{\mathrm{IMF}j}(t)} \right] - \mathrm{sgn}\left[r^{\mathrm{up}}_{\mathrm{IMF}j}(t-1) - \overline{r^{\mathrm{up}}_{\mathrm{IMF}j}(t)} \right] \right|. \tag{12}$$

ACR features can express the number of times the upper envelope of an IMF passes through its mean point, that is, the number of times the IMF's upper envelope (time domain amplitude) fluctuates significantly. If the value is large, the IMF amplitude frequently fluctuates near the mean value. For ambient audio stream recognition application scenarios, if the value is greater than a certain threshold (10 Hz or above), the data may not have obvious and meaningful hidden components and the change of the upper envelope near the mean value is only the normal fluctuation of the acoustic signal itself. If the value is less than the threshold, the data may contain significant hidden components, and one-half of the zero-crossing frequency is the frequency of the hidden components.

3.3. Ambient Audio Stream Classification. The process of audio streams classification based on heuristic mask empirical mode decomposition is shown in Figure 3. Firstly, the HM-EMD method is used to decompose the signal into the IMFS set. Then, the basic features are extracted based on the IMFS: the instantaneous frequency, instantaneous amplitude, and the hidden features of AHCFs are all composed of the feature matrix, and the feature matrix is input into the classifier to get the final recognition result. In order to verify the validity of the feature, two kinds of classifiers are selected in this paper. One is a three-layer perceptron model, whose specific structure is shown in Figure 3. The model has a three-layer structure. Sigmoid function is used as activation function for the first two layers; each layer has 500 and 250 neurons, respectively. The output layer uses a SoftMax classifier and has 10 neurons. The second is the TridentRestNet model, which consists of three branches, each of which is ResNet101. The different branches have different convolutional kernels for bottleneck modules, which use $3*3$, $5*5$, and $7*7$, respectively, in order to obtain features at different scales. Finally, all the features are fused together to give the recognition result. The experimental results show that under the two model systems, the system based on HM-EMD features still shows satisfactory results. The specific experimental results and analysis are as follows:

4. Experiments and Results

In this section, we evaluate the performance of the proposed HM-EMD method for the validity of modal separation and the audio stream classification. First, we provide details on the experimental setup which include both evaluation criteria and datasets. Second, the indexes, the results of effectiveness analysis for modal separation and acoustic scene classification methods whose performance are compared with the proposed method are provided. Finally, we compare the performance of HM-EMD with that of the baseline methods based on the experiments which are conducted on the DCASE datasets and ASVSpoof2019 and analyse the experimental results in detail.

4.1. Experimental Setup. We verify the results of this work from two aspects: the validity of modal separation and the validity of the HM-EMD features for environmental audio stream classification.

4.1.1. Validation of Modal Separation. A nonlinearity index is defined in formula (13), and it measures the stability of the decomposition results. The larger the DN is, the greater the nonlinear degree is, indicating the more unstable components; the verification data are the mixed signals of the three modes in Figure 1:

$$\mathrm{DN} = \left[\frac{1}{n}\sum_{t=1}^{n}\left(\frac{f_{IF}t - \overline{f_{IF}t}}{\overline{f_{IF}t}}\right)^2\right]^{1/2}. \qquad (13)$$

4.1.2. Validation of the Features of HM-EMD for the Classification of Audio Streams. To verify the effectiveness of designing a series of features based on HM-EMD, we use a basic HM-EMD feature matrix and a basic features + HACF matrix as the input parameters of the classifier. Specifically, the number of mask EMD reference IMFs is 20, the number of HM-EMD basic feature's dimension is 20, and HACFs is 3D, whose number of dimension is 20×3. The audio frame length is 0.5 s, the interframe overlap is 0.25 s, and the total number of dimensions is $39 \times 20 \times 3 = 2340$. The classical mel frequency cepstral coefficients are selected as the contrast features; they include 13 dimensional MFCCs and delta features. The total number of dimensions is 39, and the audio frame length is 40 ms.

After setting the characteristic parameters, we conducted the test according to the process designed in Figure 3. There are two datasets used in our experiment:

(1) TASK1A dataset of DCASE [16]: the dataset contains data on ten cities and nine devices, that is, three real devices (A, B, C) and six simulated devices (S1–S6). The dataset has good annotation, including three different types of indoor, outdoor, and traffic. It also has ten different ambient audio streams, namely, airport, shopping mall, metro, metro station, pedestrian, street traffic, tram, park and public square, and bus. The acoustic data span a total of 64 h, with 40 h used in dataset training and with 24 h used in verification. Each audio segment is 10 s long, and the sampling rate is 44.1 kHz.

(2) ASVSpoof 2019 dataset [17], which is a dataset aiming to foster the research on countermeasure to detect voice spoofing in automatic speaker verification. The dataset contains synthesized and replayed speech attacks, which are classified as logical access and physical access respectively. There are three subsets under these two tracks, namely, training set, development set, and evaluation set. Actual voice data for both tracks were collected from 107 speakers collected from the VCTK2 database, 46 male speakers, and 61 female speakers. A subset of training and development of physical access was created by simulating room acoustics, including 3 room sizes, 3 reverberation levels, and 3 speaker

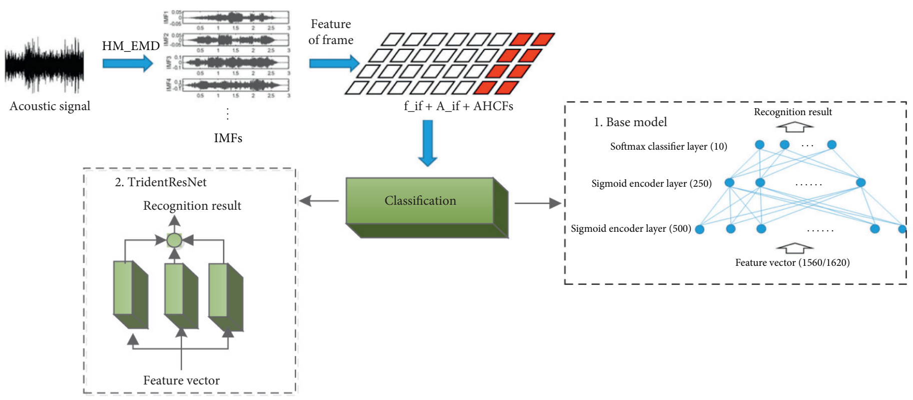


FIGURE 3: The process of audio streams classification based on heuristic mask empirical mode decomposition.

distances from the ASV microphone. In addition, there are nine recording configurations, with three recording distances to the speakers and three speakers of different qualities. Since this paper focuses on testing the feature of HM-EMD in the recognition of fake audio, the 10 neurons in the output layer of the two classifiers in Figure 3 are replaced with 2 neurons and the training the model again. At this time, the output result of the model is the detection results of fake audio. The results are evaluated by EER (equal error rate). Specific experimental results are shown in Section 4.2.

4.2. Results and Analysis

4.2.1. Effectiveness Analysis for Modal Separation. By comparing the traditional EMD results, we can see $\mathrm{DN}_{\text{HM-EMD}}$ /$\mathrm{DN}_{\text{EMD}} < 1$ for any given case. Hence, the IMF processed by the HM-EMD method has the lowest nonlinearity; that is, the IMF has a high purity and is close to the blind separation result under an ideal state. The separation result is shown in Figure 4. The features based on this high-purity IMF signal can effectively characterize the subtle changes in the signal components in the time and frequency domains. Hence, the method is suitable for all types of acoustic correlation analyses and recognition, especially for the recognition of ambient audio streams with hidden acoustic events.

4.2.2. Based on HM Feature Validity of EMD

(1) Ambient Audio Stream Classification. HACFs can be used to identify the hidden components in IMFs and are thus of great significance for ambient audio stream recognition. We verified the discrimination ability of HACFs in different scenarios (Figure 5). The figure shows the scatter projection of some hidden component features in the three-dimensional space. Even the three-dimensional features in a single IMF have a strong scene discrimination ability. HACFs show good discrimination ability among different ambient audio stream categories and thus provide technical support for subsequent ambient audio stream classification.

Figures 6 and 7, respectively, show the acoustic streams classification and recognition results of MFCC features, HM-EMD basic features and HM-EMD basic features + AHCFS features based on simple classifier and complex classifier. The simple classifier is a simple three-layer perceptron, while the complex classifier adopts the optimal classifier in the DCASE competition [18]. As can be seen from the figures, basic environmental features based on HM-EMD in the simple classifier are better than MFCC features in most scenes, and AHCFS features can effectively capture hidden information in the environment, which is improving the accuracy of audio streams classification. In the complex classification model, the improvement of model classification ability can make up the deficiency of feature representation, and the overall recognition rate has increased.

Tables 1 shows the results of acoustic streams classification. It can be seen that the HM-EMD feature is superior to the MFCC feature with different classifiers: given the basic classifier, $f_{IF} + A_{IF}$ is 6.7 percentage points higher than that in the MFCC series. After the addition of HACFs, the recognition rate increases by 17.4 percentage points. This result is close to the classification accuracy of the RESNET network with a 32M model size in the DCASE competition [19], while the simple model used in this paper is only 225K. In a complex classification model, the improvement of model classification can make up for the lack of features to some extent. However, in this case, $f_{IF} + A_{IF}$ + HACFs still improves the accuracy by 1.3%, and the recognition result reaches 75.7%. The $f_{IF} + A_{IF}$ feature can provide instantaneous characteristics in time-frequency domain and HACFs represent the statistical characteristics in time-frequency domain. The combination of the two can help improve the accuracy of the classification of environmental audio streams.

(2) Fake Audio Detection. Table 2 presents the fake audio detection results based on HM-EMD features. It can be seen that the forged audio based on LA is easier to be identified than the forged audio based on PA, and the HM-EMD feature has a more effect on the fake audio of LA attack. After adding the hidden information feature AHCF, the detection error rate of forged audio was reduced by 5.61% and 6.11%, respectively. This is because the LA

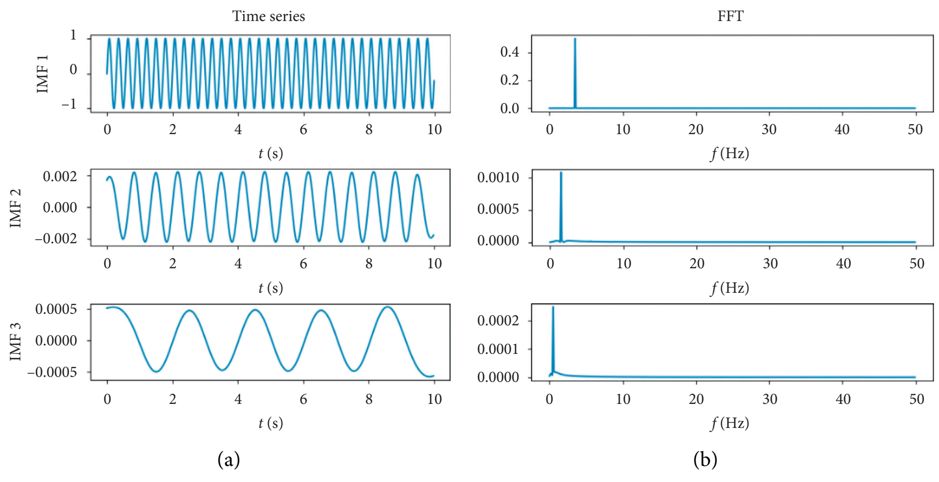


Figure 4: HM-EMD results of $x_1(t)$.

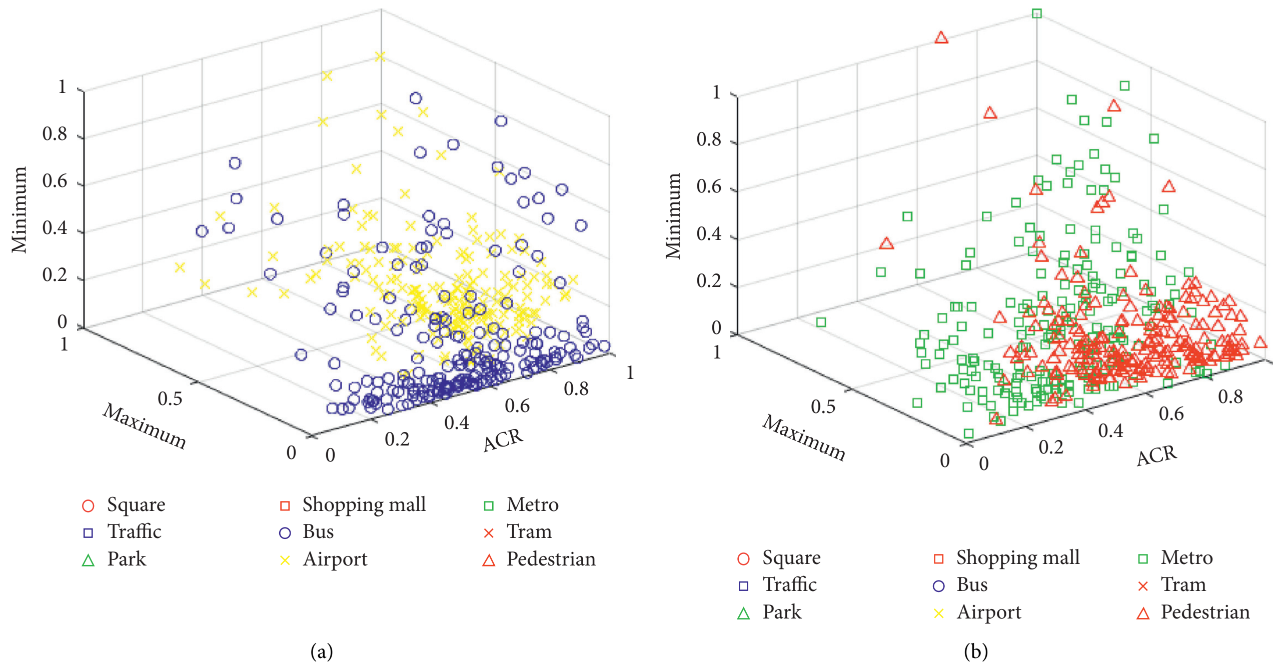
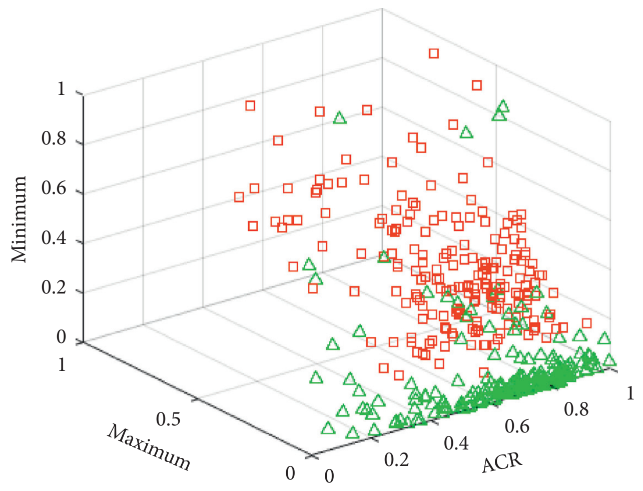
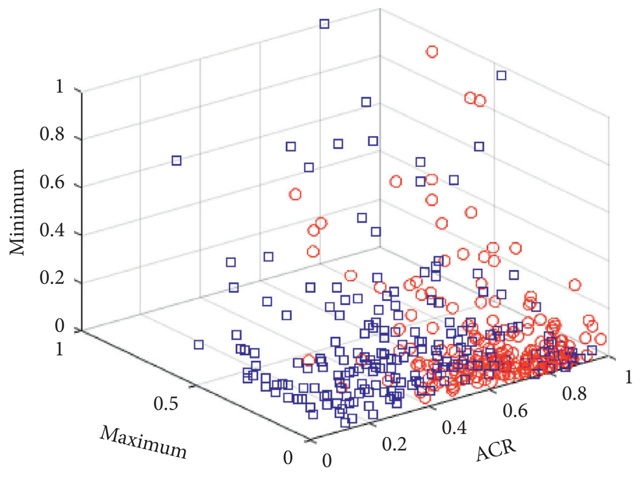


Figure 5: Continued.



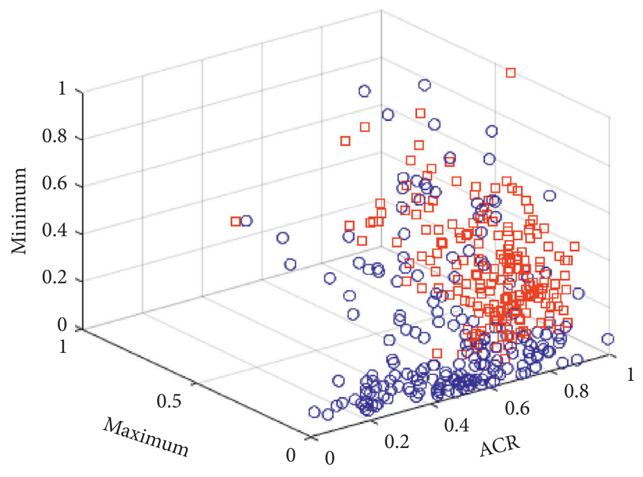
○ Square □ Shopping mall □ Metro
□ Traffic ○ Bus × Tram
△ Park × Airport △ Pedestrian

(c)



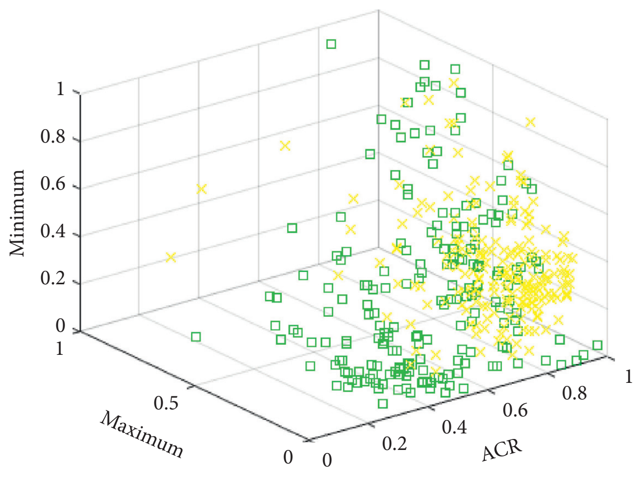
○ Square □ Shopping mall □ Metro
□ Traffic ○ Bus × Tram
△ Park × Airport △ Pedestrian

(d)



○ Square □ Shopping mall □ Metro
□ Traffic ○ Bus × Tram
△ Park × Airport △ Pedestrian

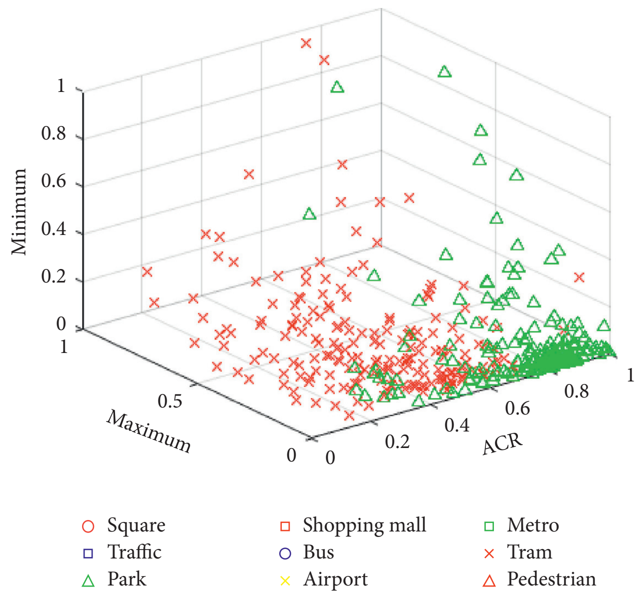
(e)



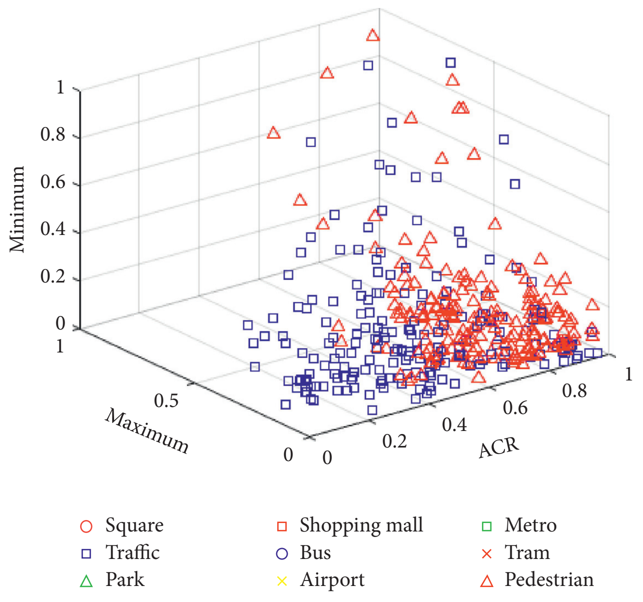
○ Square □ Shopping mall □ Metro
□ Traffic ○ Bus × Tram
△ Park × Airport △ Pedestrian

(f)

FIGURE 5: Continued.



(g)



(h)

FIGURE 5: HACFS feature distribution in 3D space. (a) IMF1 of bus and airport. (b) IMF1 of metro and pedestrian. (c) IMF1 of park and shopping mall. (d) IMF1 of square and traffic. (e) IMF2 of bus and shopping mall. (f) IMF2 of metro and airport. (g) IMF2 of tram and park. (h) IMF2 of traffic and pedestrian.

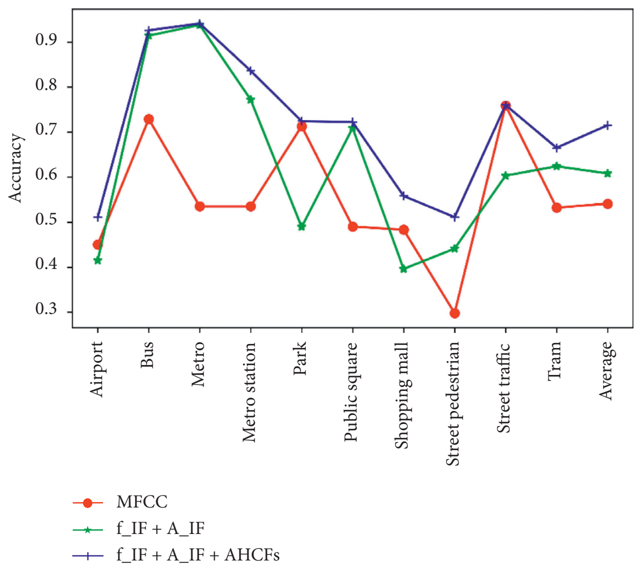


FIGURE 6: Acoustic streams' classification results of DCASE based on the simple classifier.

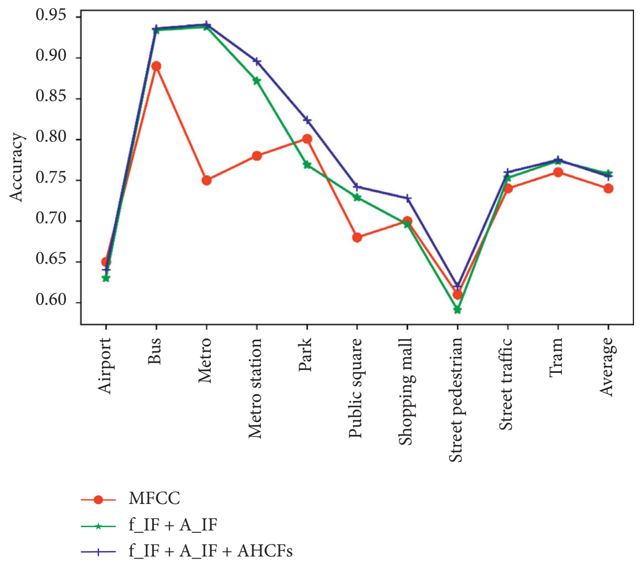


FIGURE 7: Acoustic streams' classification results of DCASE based on TridentresNet.

ignores background sound in the process of synthesizing the fake audio. In this case, the addition of captured audio background features greatly reduces the detection error rate. It can also be seen from the table that the HM-EMD feature can reduce the error rate of fake audio detection in both simple and complex models, which also proves the effectiveness of the feature extraction method proposed in this paper.

Table 1: Average results of acoustic streams classification based on DCASE dataset.

Classifier	MFCC (accuracy %)	$f_{IF} + A_{IF}$ (accuracy %)	$f_{IF} + A_{IF}$ + AHCFs (accuracy %)
TridentResNet_DevSet	73.7	74.5	75.0
TridentResNet_EvalSet	73.7	74.5	75.0
TridentResNet_Ensemble	74.2	75.0	75.5
TridentResNet_Weighted_Ensemble	74.4	75.2	75.7
3L neural network	54.1	60.8	71.5

Table 2: Fake audio detection results based on ASVSpoof 2019 dataset.

Classifier	Attack type	MFCC (EER %)	$f_{IF} + A_{IF}$ (EER %)	$f_{IF} + A_{IF}$ + AHCFs (EER %)
TridentResNet	LA	7.06	2.82	1.45
	PA	9.55	8.51	7.14
3L neural network	LA	9.95	4.98	3.84
	PA	11.83	11.79	9.44

5. Conclusions

Aiming at the problem of audio fraud existing in the network, this paper proposes a method of feature extraction of hidden information in audio streams based on HM-EMD. Because the audio background information is difficult to be forged, the basic features of audio streams and the characteristic HCFS of hidden information are constructed for fake audio streams based on stable IMFs decomposed by the HM-EMD method. The experimental results show that the HM-EMD-based features have richer characterization ability for hidden acoustic events than MEL cepstrum features and can improve the accuracy of scene classification and fake audio detection. However, since the HM-EMD decomposition process needs to calculate the mask signal according to the structure of the signal itself and use the mask signal to separate the aliasing signal, the algorithm complexity is increased compared with the classical EMD algorithm. Therefore, in the subsequent work, we will consider the idea of coevolutionary framework to optimize the algorithm [20]. At same time, the relationship between the HM-EMD feature system and different hidden acoustic events will be the further exploration point, so as to achieve accurate hidden acoustic event markers in audio streams of different levels and time scales. In general, the feature extraction of audio streams based on HM-EMD is helpful to detect fake audio and provides a new research idea for solving network audio spoofing.

Data Availability

The data used in this study are the public dataset from DCASE challenge (http://dcase.community/challenge2020/task-acoustic-scene-classification and https://datashare.ed.ac.uk/handle/10283/3336).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Jiu Lou and Decheng Zuo conceived and designed the study. Zhongliang Xu performed the simulations. Hongwei Liu reviewed and edited the manuscript. All authors read and approved the final manuscript.

Acknowledgments

This work was in part supported by the National Key Research and Development Program (2018YFC0830602).

References

[1] A. Jati, C.-C. Hsu, M. Pal et al., "Adversarial attack and defense strategies for deep speaker recognition systems," *Computer Speech & Language*, vol. 68, 2021.

[2] X. R. Li, S. L. Ji, C. M. Wu et al., "Survey on deepfakes and detection techniques," *Journal of Software*, vol. 32, no. 2, pp. 496–518, 2021.

[3] T. Chen, A. Kumar, P. Nagarsheth et al., "Generalization of audio deepfake detection," in *Proceedings of the Odyssey 2020 the Speaker and Language Recognition Workshop*, pp. 132–137, Tokyo, Japan, November 2020.

[4] R. K. Das, J. C. Yang, and H. Z. Li, "Long range acoustic features for spoofed speech detection," in *Proceedings of the Annual Conference of the International Speech Communication Association*, pp. 1058–1062, Graz, Austria, September 2019.

[5] M. Todisco, X. Wang, V. Vestman et al., "ASVSpoof 2019: future horizons in spoofed and fake audio detection," in *Proceedings of the Annual Conference of the International Speech Communication Association*, pp. 1008–1012, Graz, Austria, September 2019.

[6] M. Todisco, H. Delgado, and N. Evans, "A new feature for automatic speaker verification anti-spoofing: constant Q cepstral coefficients," in *Proceedings of the Odyssey 2016 the Speaker and Language Recognition Workshop*, pp. 283–290, Bilbao, Spain, June 2016.

[7] R. K. Das, J. C. Yang, and H. Z. Li, "Long range acoustic and deep features perspective on ASVspoof 2019," in *Proceedings of the 2019 IEEE Automatic Speech Recognition and*

Understanding Workshop, pp. 1018–1025, Singapore, December 2019.

[8] Z. Z. Wu, T. Kinnunen, E. S. Chng et al., "A study on spoofing attack in state-of-the-art speaker verification: the telephone speech case," in *Proceedings of the 2012 Conference Handbook—Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, Hollywood, CA, USA, December 2012.

[9] A. Gomez-Alanis, A. M. Peinado, J. A. Gonzalez et al., "A light convolutional GRU-RNN deep feature extractor for ASV spoofing detection," in *Proceedings of the Annual Conference of the International Speech Communication Association*, pp. 1068–1072, Graz, Austria, September 2019.

[10] H. Zeinali, T. Stafylakis, G. Athanasopoulou et al., "Detecting spoofing attacks using VGG and SincNet: BUT-omilia submission to ASVspoof 2019 challenge," in *Proceedings of the Annual Conference of the International Speech Communication Association*, pp. 1073–1077, Graz, Austria, September 2019.

[11] R. J. Li, M. Zhao, Z. Li et al., "Anti-spoofing speaker verification system with multi-feature integration and multi-task learning," in *Proceedings of the Annual Conference of the International Speech Communication Association*, pp. 1048–1052, Graz, Austria, September 2019.

[12] S. H. Mankad, S. Garg, M. Patel, and H. Adalja, "Investigating feature reduction strategies for replay antispoofing in voice biometrics," in *Proceedings of the 8th International Conference on Pattern Recognition and Machine Intelligence*, Tezpur, India, December 2019.

[13] S. H. Mankad and S. Garg, "On the performance of empirical mode decomposition-based replay spoofing detection in speaker verification systems," *Progress in Artificial Intelligence*, vol. 9, no. 4, pp. 325–339, 2020.

[14] R. Deering and J. F. Kaiser, "The use of a masking signal to improve empirical mode decomposition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 485–488, Philadelphia, PA, USA, March 2005.

[15] H. Li, Z. Li, and W. Mo, "A time varying filter approach for empirical mode decomposition," *Signal Processing*, vol. 138, pp. 146–158, 2017.

[16] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," Technical report, Tampere University of Technology, Tampere, Finland, 2018.

[17] X. Wang, Y. S. Junichi, T. Massimiliano et al., "ASVspoof 2019: a large-scale public database of synthesized, converted and replayed speech," Technical Report, Cornell University, Ithaca, NY, USA, 2020.

[18] T. Heittola, A. Mesaros, and T. Virtanen, "TAU urban acoustic scenes 2020 mobile, development dataset," *Tampere University, Tampere, Finland*, 2020, Technical Report.

[19] H. J. Shim, J. H. Kim, J. W. Jung et al., "Audio tagging and deep architectures for acoustic scene classification: Uos submission for the DCASE 2020 challenge," 2020, http://dcase.community/documents/challenge2020/technical_reports/DCASE2020_ Shim_ 120.pdf.

[20] W. Deng, S. F. Shang, X. Cai et al., "Quantum differential evolution with cooperative coevolution framework and hybrid mutation strategy for large scale optimization," *Knowledge-Based Systems*, vol. 224, pp. 1–14, 2021.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9965157, 14 pages
https://doi.org/10.1155/2021/9965157

Research Article

DAM-SE: A Blockchain-Based Optimized Solution for the Counterattacks in the Internet of Federated Learning Systems

Shichang Xuan, Ming Jin, Xin Li, Zhaoyuan Yao, Wu Yang, and Dapeng Man

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

Correspondence should be addressed to Dapeng Man; mandapeng@hrbeu.edu.cn

Received 26 March 2021; Accepted 21 June 2021; Published 1 July 2021

Academic Editor: Muhammad Shafiq

The rapid development in network technology has resulted in the proliferation of Internet of Things (IoT). This trend has led to a widespread utilization of decentralized data and distributed computing power. While machine learning can benefit from the massive amount of IoT data, privacy concerns and communication costs have caused data silos. Although the adoption of blockchain and federated learning technologies addresses the security issues related to collusion attacks and privacy leakage in data sharing, the "free-rider attacks" and "model poisoning attacks" in the federated learning process require auditing of the training models one by one. However, that increases the communication cost of the entire training process. Hence, to address the problem of increased communication cost due to node security verification in the blockchain-based federated learning process, we propose a communication cost optimization method based on security evaluation. By studying the verification mechanism for useless or malicious nodes, we also introduce a double-layer aggregation model into the federated learning process by combining the competing voting verification methods and aggregation algorithms. The experimental comparisons verify that the proposed model effectively reduces the communication cost of the node security verification in the blockchain-based federated learning process.

1. Introduction

With the continuous expansion of the scale of Internet of Things and the rapid development of artificial intelligence, the amount of data generated is growing at an unprecedented speed. If these resources distributed in the edge of the Internet of Things are effectively used, this will greatly improve the efficiency of machine learning and reduce the cost of machine learning. Therefore, how to use the data generated by Internet of Things devices efficiently becomes a hot topic. However, with the deepening of research, how to transfer the value of data under the premise of ensuring data security has become a key problem to be solved. On the one hand, the centralized management of the existing Internet of Things may lead to data centralization, which leads to the problem of single point failure and affects the smooth operation of the whole network. On the other hand, in the process of information transmission, due to the openness of wireless network, wireless signals between transmission devices are easily eavesdropping, interfering and shielding, and being attacked by DDoS and Sybil, which will affect the security of privacy data. In view of the above problems, the combination of blockchain and federated learning advantages and application in the field of Internet of Things can effectively improve the security of the Internet of Things.

Blockchain is an emerging and rapidly developing technology. Due to its wide commercial application, many research directions have been formed. Through the combination of various technologies, we have obtained unique advantages and capabilities and greatly expanded some application fields and functions. Blockchain can effectively integrate resources to form a perfect architecture and promote the development of machine learning technology [1]. Some examples include helping doctors make more accurate diagnosis [2] and providing more humanized and intelligent services for the Internet of Things [3]. At present, researches on federated learning based on blockchain are increasing. The main research direction in this field is to use blockchain to replace the central node in federated learning and use the decentralization of blockchain to overcome the privacy leakage and single point of failure problems caused by the node.

In traditional scenarios, enterprises or institution training models need to collect, store, and process a large amount of data. This means higher network, storage, and computing capabilities, as well as training costs. In the process of data transmission and sharing, the following challenges are encountered: (i) "Data island" phenomenon, (ii) stricter security regulations, and (iii) the data storage capacity that cannot meet the practical application requirements. To overcome these challenges, federated learning (FL) came into being. In FL process, the training node uses the local data set to train a local model. Then, the training node sends the parameters of the local model to the central coordinator, which aggregates multiple local models to get a global model. Finally, the training node downloads the parameters of the global model to update the local model and then trains on the local data set again, so that *n* rounds of iterations are carried out until the global model converges. FL can cooperate with or learn in depth on the Internet of Things edge network to ensure data security because the data set does not need to be migrated throughout the learning process. However, due to the complexity of the Internet of Things, different edge devices show different quality and stability in the learning process. This leads to different requirements for communication cost, privacy protection, and resource allocation, which increases a certain bottleneck for the wide spread of federated learning. Some researchers have proposed some schemes, such as verification process and algorithms, to overcome the "free-rider" and "model poisoning" attacks in the federated learning process. However, the current solution needs to verify and audit all local models one by one, which not only results in the waste of computing power but also increases the communication cost of the whole training process.

The main contribution of this paper is to compare existing verifiable federated learning frameworks based on blockchain and propose a double-layer aggregation model based on security evaluation to address its high communication cost. At the same time, the proposed incentive mechanism ensures that rational workers can gain the maximum benefit by remaining honest. The combination with the incentive model allows the proposed model to reduce the communication cost of training without relying on any heavy encryption and special hardware and to defend against poisoning attacks and free-rider attacks.

The rest of the paper is organized as follows. We discuss the existing works in more detail and compare them with our work in Section 2. The double-layer polymerization model based on safety evaluation is described in Section 3. Section 4 gives the conclusion.

2. Related Works

Federated learning can train the local model and aggregate the global model on the premise that the private data does not leave the local, to protect the security of the private data. However, there are still some defects in the model aggregation and the benign incentive of participating nodes in federated learning, which leads to a series of researches.

In terms of security of model aggregation, Fu et al. [4] proposed a privacy-protected verified federated learning (VFL) algorithm. This algorithm used Lagrange interpolation to carefully set the interpolation points verifying the correctness of the polymerization gradient, which preserved the safety of the aggregation model. If no more than $n-2$ of n participants collude with the aggregation server, VFL could guarantee the encrypted gradients of other participants not being inverted.

In decentralized federated learning, Kim et al. [5] proposed a blockchain federated learning (BlockFL) architecture. Since the local training results included a verification process, BlockFL overcame the single point of failure and expanded its federated scope to untrusted devices in public networks. Besides, it promoted the combination of more equipment with more training samples by providing rewards proportional to the number of training samples. Bao et al. [6] proposed a centered, public-audited, and healthy federated learning ecosystem called FLchain in trust and incentive. In FLchain, blockchain is used to replace the traditional federated learning central coordinator. With the tamper-resistant nature of blockchain, the behavior of participating nodes can be trusted, so that benign and malicious participating nodes can be identified and incentivized and punished accordingly. Majeed and Seon [7] proposed a new blockchain architecture. The concept of channels is used to learn multiple global models of this architecture. The local model parameters for each global iteration are stored as blocks in a channel-specific ledger. Zhang et al. [8] proposed a decentralized, collaborative privacy protection training method based on a multizone blockchain system for medical image analysis. This method allowed researchers from different medical institutions to collaborate when training machine learning models without sharing sensitive patient data.

In the area of security collaboration, Yin et al. [9] developed a federated learning-based security data collaborative (FDC) mechanism for handling the safety collaboration of multiparty data in the IoT environment. The blockchain was used to record the multipartial interaction content addressing the privacy and security issues of the IoT data. Federated learning was used to solve the problem of large-scale multiparty security collaboration in the IoT. Kim and Hong [10] et al. proposed a local learning weighting method based on node identification, and the experimental results show that the method proposed in this paper performs better in terms of learning speed and stability compared to the traditional federated learning. Martinez et al. [11] addresses the issues of data privacy, security, and fair rewards in distributed machine learning using blockchain and federated learning. An in-depth workflow, off-chain record database that can be used in conjunction with blockchain and an architecture for scalable recording and rewarding of gradients are proposed. Zhang et al. [12] proposed a blockchain-based federated learning method to detect equipment failures in the IoT. This method enables the authentication integrity of client data. To solve the data heterogeneity problem in failure detection, a novel centroid weighted joint average algorithm called CDWFEDAVG is proposed. Finally, to motivate customers to participate in the federated learning process, an incentive mechanism is designed based on the customer data used in local model training.

In terms of privacy protection, Lu et al. [13] design a secure data-sharing architecture supporting zone chains. By adopting federated learning, data sharing was expressed as a machine learning problem. They also showed that the data privacy could be protected by sharing data models rather than the actual data. Zhao et al. [14] designed a federated learning system using credit mechanisms. This system allows household appliance manufacturers to predict customer needs and consumption behaviors based on machine learning models trained using customer data. They also proposed a new standardization technology, which achieved a higher test accuracy than the bulk standardization while retaining the extraction characteristics privacy of each participant data. Besides, by using differential privacy, the opponent could be prevented from inferring the customer's sensitive information. Yu et al. [15] proposed a new privacy-preserving federated learning scheme. Based on the trusted execution environment (TEE), the training Integrity Protocol of the scheme was designed. The protocol can detect causal attacks and ensure the integrity of the deep learning process. Kim et al. [16] proposed a blockchain federated learning (BlockFL) scheme. This scheme uses the method of combining blockchain technology and federated learning to solve the problem that the central coordinator which the centralized federated learning system depends on is vulnerable to attack. Wang et al. [17] proposed a new secure decentralized multiparty learning system based on blockchain technology. The authors design two types of Byzantine attacks in the system and design secure off-chain sample mining and on-chain sample mining schemes to resist the attacks.

At present, there are not particularly many blockchain-based federated learning technologies and platforms, and most of the research in this field uses blockchain instead of central servers to ensure the security and accuracy of federated learning aggregation. Meanwhile, the incentive mechanism of blockchain can attract more nodes to participate in training. However, there is less research related to node security verification, which deserves in-depth study.

3. The Double-Layer Aggregation Model Based on Safety Evaluation

In this section, we introduce the two main models of the current blockchain-based verifiable federated learning framework and subsequently introduce the double-layer aggregation model proposed in this paper with its security evaluation index and finally present our comparative experiments and the analysis of the results.

3.1. Comparison of Verified Federated Learning Frameworks. There are currently two main models of blockchain-based verifiable federated learning frameworks: (i) the centralized verification model of the verification set and (ii) the distributed verification model of all submodels.

An example of the first model is the EOS blockchain of Martinez et al. [11], which proposed a new concept based on the data segmentation of the edge node customizing the verification data set called class sample verification error schemes (CSVES), as shown in Figure 1. This method is recommended to be used before the start of training. The collection of all available classes is defined as $C = \{C_1, C_2, \ldots, C_p\}$. For edge node $D \in D$, the article puts the collection C_D as a collection of all categories of data. Then, $C_D \subseteq C$, since there might be a class $C_i \in C$, making all local data points $d \in d$, where $d \notin C_i$. D sends a data set C_D to O and receives a validation set, C_D. The data set of the verification set C_D is only selected from the chain data set of O. Once the verification set is received, D starts with the local training data set d and the received verification set training model T_k. During the training, the model applies the verification set to the training model and records the verification errors. If the number of verification errors is reduced, the model is considered to be improved. At the end of the training, D sends the authentication error information and other parameters of the call function *UploadGradient ()*. Reference [11] improved on this feature to observe the overall trend of verification errors during model training. If the verification error is reduced, δ is a valuable and valid gradient update, and the model is rewarded based on its data cost n.

Toyoda et al. [18] proposed another competitive model update method called IABFL, as shown in Figure 2. IABFL was a low-cost approach achieving the expected goals in the event of reasonable action of participants. The main focus was on federated learning to introduce duplicate competition, which allowed the rational workers to follow the agreement and maximize their profits. Each worker selected in a particular round picked the top update model submitted by the last round of worker and used it to update their model.

EOS and IABFL are able to validate the child model updated by the edge node participating in the training to prevent useless or malicious models from joining to the global model. In the EOS, each training round requires participants to claim the validation data set from the training process on the blockchain based on the local data set.

The size of the validation data set is not as large as the local data set, but, in the overall framework, it requires the blockchain to send different validation sets to each edge node participating in the training. The number of edge nodes in a typical federation learning framework is measured in tens of thousands and multiplied by the total number of rounds in the whole training process, the communication consumption of the scheme will be very large, and the communication cost of training a completed model will be tens or even hundreds of times higher than that of unverified federation learning.

In the IABFL, the submodel of each node is transmitted between blockchains as the communication data during the verification process. The amount of submodel data is not voluminous compared to the verification data set; hence, the communication cost of the program will be much lower than that of the EOS.

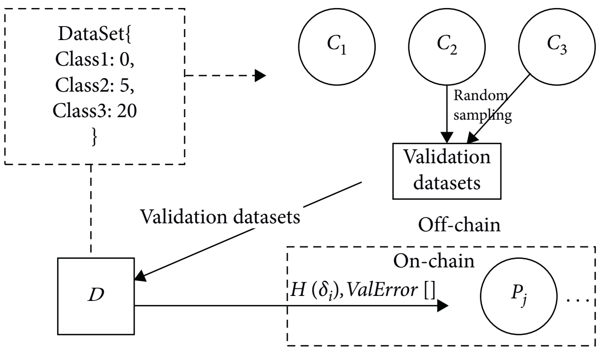


FIGURE 1: Class sample verification error schemes.

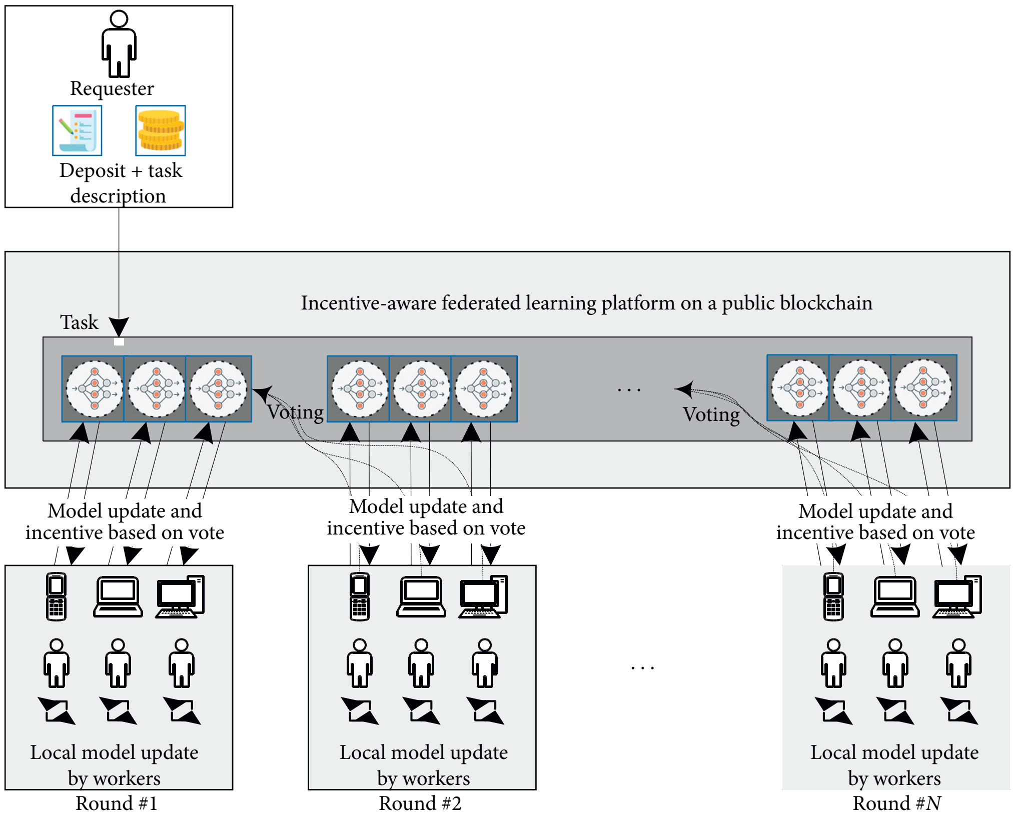

Requester
Deposit + task description
Incentive-aware federated learning platform on a public blockchain
Task
Voting
Voting
...
Model update and incentive based on vote
Model update and incentive based on vote
Model update and incentive based on vote
Local model update by workers
Round #1
Local model update by workers
Round #2
...
Local model update by workers
Round #N

FIGURE 2: Class sample verification error schemes.

In this model, it is assumed that n worker members are involved in training each round, and the traffic created by transmitting a single model parameter in the network is t. First, n worker members submit a local model to the block and synchronize the child model of all other nodes on the chain. This process requires a data traffic of C_1, which can be expressed as

$$C_1 = n \times (n - 1) \times t. \quad (1)$$

Assuming that the amount of data consumed by a single vote is 1, n worker participating in training completes voting on the chain and synchronizes the data traffic that needs C_2; that is,

$$C_2 = n \times (n - 1). \quad (2)$$

The data traffic created by n worker in the IABFL is denoted as C_{pre}, which can be calculated as

$$C_{\text{pre}} = C_1 + C_2 = n \times (n - 1) \times t + n \times (n - 1). \quad (3)$$

Through the analysis of communication cost, it can be found that the current work has solved the problem of security verification of nodes relatively well, but it brings a significant increase in communication cost, and the effect is doubtful in practical application, so this chapter wants to propose a corresponding solution to the problem of high communication cost.

3.2. The Double-Layer Polymerization Model. In the traditional blockchain-based federated learning process, training is vulnerable to model poisoning attacks or free-riding attacks. The current verifiable federated learning will add a large amount of data communication on the basis of the original training to verify the quality of nodes or data in various ways to solve the above problems. However, the sharp increase in communication costs will reduce the willingness of each node to participate, so we propose a step-by-step federated learning platform based on blockchain with a verification mechanism design.

The key idea behind our platform is to introduce repeated models to update the competition design, and through incentive mechanisms any rational worker can work hard and abide by the agreement to maximize their profits. The proposed design will naturally enable rational workers to act honestly without any heavy encryption and special hardware. As shown in Figure 3, in a specific round, all the nodes participating in the training are divided into multiple small clusters. The submodels of several nodes in a small cluster are first partially aggregated into a step-by-step model, and each child node will select the upper A round of the best k model updates submitted by a small cluster and update its own model based on these updates. The reason is that the reward for workers in the previous round depends on the results of the voting. The motivation for choosing the model with the best k models is that its model updates will have more chances to be voted in the next round, which means that they will get more return. The workers in the next round still cannot be destroyed, because their models are also competed and voted on by the workers in the next round. In the following, we will discuss the system model and the mechanism process in detail.

The symbols used later are described in Table 1.

There are four roles in the system: administrator, requester, worker, and consensus node. Table 2 lists the role information of each participant. The role of the administrator is to deploy a series of smart contracts [19] on a public blockchain, such as Ethereum [20], and to register requesters and workers to the platform upon request. It is assumed that the participants know how to access the location of the smart contracts through a forum or website. The requester may have neither the data for training nor the equipment for training deep learning models. The worker, on the other hand, needs to have both data for training and equipment for training the deep learning model. Any type of data can be processed on this platform, such as images, text, and audio. Enter the data set of worker i for task t. Subscript t is omitted because the next are for a specific task t. It is assumed that the data sets owned by workers for a specific task are independent and homogeneously distributed. This assumption is natural because a requester submitting a model for a specific task, such as a deep learning model for identifying cats in pictures, would require that only workers with data sets specific to that task can join.

The platform consists of seven procedures: user registration, task release, task joining, task start, model update, reward assignment, and task completion. In this article, Ethereum is used as the blockchain, which is one of the most popular cryptocurrencies that support intelligent contracts. However, the proposed model is applicable to any other technology with intelligent contract support.

3.2.1. User Registration. Administrators need to register all participating users on the platform based on their requests. Each user must share their Ethereum address with the administrator for receiving tasks and rewards, besides declaring whether to register as a requester or worker. After the registration is completed, the requester can release the FL training task on the blockchain, and the worker can join the task to update the model and get the corresponding reward.

3.2.2. Task Release. Any user registered as a requestor can issue federated learning training tasks through a smart contract. To do that, the requester must specify the following:

(1) Model description, for example, loss functions, data formats, learning rates, layers, unit numbers, and activation functions
(2) The parameters, for example, the training period, safety evaluation index, start time, the number of workers, and the total rewards
(3) Deposits of the total rewards, D, which are equal to $r \times N$

3.2.3. Task Joining. After the requester posts the task, event notifications will be sent to all registered workers via the Ethereum's event processing function. Each worker will then decide whether to participate in that task. If the worker decides to join, the intelligent contract should be called before the task begins. Based on the requirements, the intelligent contract can only be called when the caller is registered as a worker; otherwise, the abort code is executed. From the perspective of code implementation, the EMI address of a worker is stored in an array.

3.2.4. Task Starts. After the task application period, the requester enrolls in the group of workers W_t joining the task t. Then, they select the number of the model updates N and the number of worker members participating in each round. However, the requester should not disclose in advance to the worker the number of rounds N to be used for model updates. It is necessary to show the worker at the end of the N-wheels. The cause of this will be detailed later. Besides, the requester needs to introduce the federated learning model parameters into ω_0 and submits them into the blockchain. The requester can use any algorithm to initialize the model [21].

3.2.5. Model Update. After model training, each round of worker k is randomly selected among all members registered with the intelligent contract. Then, the number of individuals in each cluster is calculated according to the safety evaluation algorithm, which will be detailed in the next sections. Each worker gets the local aggregation model parameters of the previous round of each cluster from the blockchain and verifies and votes them. Then, they calculate the global model for the model update based on the selected top model. Finally, each worker is trained based on the local

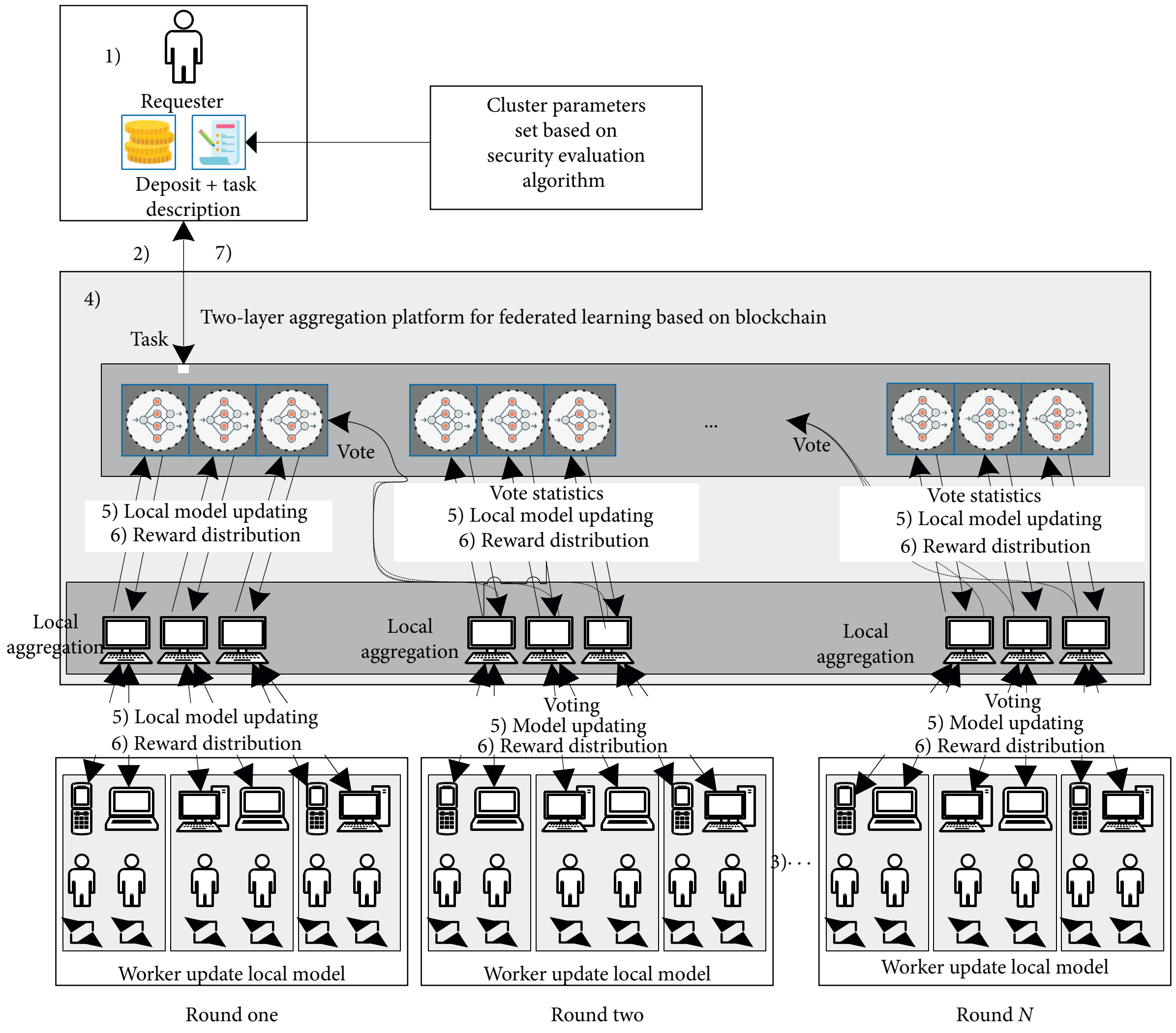

1)
Requester
Deposit + task description
Cluster parameters set based on security evaluation algorithm
2)
7)
4)
Two-layer aggregation platform for federated learning based on blockchain
Task
Vote
...
Vote
Vote statistics
5) Local model updating
6) Reward distribution
Local aggregation
Voting
5) Model updating
6) Reward distribution
3)...
Worker update local model
Round one
Round two
Round N

Figure 3: The double-layer FL aggregation platform based on blockchains.

Table 1: Symbols table.

Symbol	Paraphrase
W	The entire collection of workers
i	Worker's index
e	Index of each round
a	The number of top model updates selected by the staff in the next round
K	Total number of workers participating in task t
C	Percentage of workers per round
k	Number of workers selected in each round, $k = \max(K \cdot C, 1)$
d_i	Data set owned by worker i in round e
B	Batch size in deep learning
η	Learning rate in deep learning
s	The security value
g	Number of clusters in federated learning
$\tau(\cdot)$	Loss function, such as the mean square error in deep learning
$\text{split}(d, B)$	Function to randomly divide data set d into batch B

data set to derive the submodel for this round and submits it to the blockchain consensus node of the cluster. The consensus node locally aggregates the submodels of all workers of the cluster according to the average aggregation algorithm to obtain the local model parameters of the cluster and submits them to the chain together with the voting results of each node. The algorithm for model update is shown as Algorithm 1.

In lines 1–6 of the algorithm, the main task is to select the best top models, a, for voting and provide them for use in subsequent model aggregations. This is done as follows: if it is not the first round at the beginning of the training task, each worker uses the local data set to validate the local aggregation models of the g clusters from the previous round and selects a model that they consider the best for voting; otherwise, this step is skipped. In lines 7–11 of the algorithm, the role is to compute the underlying global model for this training round. This is done by using the initialization parameter ω_0 provided by the requester as the

Table 2: The role of each participant.

Participant	Task	Availability of data	Availability of equipment for model training
Administrator	Deploy smart contracts on the public blockchain and register requesters and workers to smart contracts upon request	—	—
Requester	Submit a training task to obtain a trained model	Unnecessary	Unnecessary
Worker	Train the task model submitted by the requester for the reward	Yes	Yes
Consensus node	Local aggregation, synchronization of blockchain information, and distribution of rewards to submodels in the cluster	Unnecessary	Unnecessary

```
Input: model updates submitted by all clusters in round $e-1$, $\{\omega_{i,e-1}\}_{i\in g}$ (when $e\geq 2$) or $\omega_0$ (when $e=1$)
Output: trained model parameters $\omega_i$, Voting results $M_{i,e}$
/*1. Use local data to select the best $a$ model to update (except the first round)*/
(1) if $e\geq 2$ then
(2)     for $m\in g$ do
(3)         $l_m = 1/|d_i|\sum_{j\in d_i}\tau(X_j, y_j; \omega_{m,e-1})$
(4)     end
(5)     $M_{i,e}\longleftarrow$ chose $a$ models whose $l_m$ is the smallest.
(6) end
/*2. Aggregation with $a$ models value */
(7) if $e == 1$ then
(8)     $\omega'_{i,e}\longleftarrow\omega_0$
(9) else
(10)    $\omega'_{i,e}\longleftarrow 1/a\sum_{m\in M_{i,e}}\omega_{m,e-1}$
(11) end
/* Update the model with local data */
(12) $\beta\longleftarrow$ split$(d_i, B)$
(13) $\omega_{i,e}\longleftarrow\omega'_{i,e}$
(14) for each local epochs $E$ do
(15)     for each batch $b$ in $\beta$ do
(16)         $\omega_{i,e}\longleftarrow\omega_{i,e}-\eta\nabla\tau(\omega_{i,e}, b)$
(17)     end
(18) end
(19) return $\omega_{i,e}$, $M_{i,e}$
```

Algorithm 1: Model update of worker i in round e.

parameter of the global model if it is the first round at the beginning of the training task; otherwise the a local models selected in lines 1–6 of the algorithm are averaged and aggregated to obtain the global model. In lines 12–18 of the algorithm, each worker performs local model training using the local data set based on the global model computed in lines 7–11 of the algorithm to train the submodel parameters for that round. It is worth noting that there are E rounds of local training, and the data set in each round is not the entire local data set, but the data set is first randomly divided into b batches before training, and each round of local training uses one of these batches.

3.2.6. Reward Assignment. As shown in the algorithm in the model update, in the submission phase of the model update, each worker in round e votes for the first g local models (only one vote can be cast for a model). Based on the combined votes, the smart contract calculates the number of votes received by each cluster in round $e-1$. Based on the result of the number of votes, the reward is assigned to each cluster by $r_1\geq r_2\geq\cdots\geq r_k\geq 0$. Thus, the cluster with the most votes receives a reward of r_1, the cluster with the second most votes receives a reward of r_2, and so on. Each cluster distributes the profit according to the amount of data involved in training by the child nodes in each cluster based on the respective rewards obtained.

The total reward of each round is set to r, and the profit relationship between each cluster is given as

$$\sum_{j\in[1,k]} r_j = r. \tag{4}$$

3.2.7. Task Completion. After model update and reward assignments are repeated $N-1$ times, since there is no training task as well as workers in the next round, it is not possible to vote on the model updates completed by the workers in the last training round N. Therefore, the rewards from the last round of tasks are equally distributed to all workers involved in the training. However, this may negatively affect the working of the worker, because if the worker knows that they are selected to participate in the last round, they can get a reward only if they send one of the previous model updates. If that happens, the motivation of the first few workers will reduce, since their model updates may not receive the correct vote in the next round. Therefore, to ensure effective model training, the worker must not know whether they are in the last round of the training. So, the requester needs to reveal N to all workers at the end of the Nth round. [22]. In order not to let the worker guess N value from the remaining deposits, the requester needs to provide a deposit D larger than the actual total reward $N \times r$ before task initialization. Furthermore, after the worker completes all tasks, the requester asks them to return their excess deposits.

3.3. Safety Evaluation Index. Assuming that there are n training workers and the consensus nodes g connect to the blockchain in each round, the amount of communication consumed to transmit a single model parameter in the network is t. Firstly, the training worker members obtain the model parameters from the blockchain for the local aggregation of each cluster in the previous round. This process requires C_1 data traffic, which is calculated as

$$C_1 = n \times g \times t. \tag{5}$$

Each worker then updates the local model and submits the trained model parameters and its own voting results to the consensus node of the cluster it belongs to, and this process needs C_2 data traffic, which is calculated as follows:

$$C_2 = n \times (t + 1). \tag{6}$$

Finally, the consensus node g aggregates the child node model in the cluster into a local model. The local model and voting results of the cluster submitted to the block synchronize the local model of all other consensus nodes on the chain. This process requires C_3 data traffic [23] as follows:

$$C_3 = g \times (g - 1) \times (t + 1). \tag{7}$$

In this article, the data traffic consumed by n training workers in the model architecture C_{step} is calculated as

$$C_{\text{step}} = \sum_{j=1}^{3} C_j = n \times g \times t + n \times (t + 1) + g \times (g - 1) \times (t + 1). \tag{8}$$

According to the previous communication cost analysis, it is known that the amount of data traffic in the IABFL scheme is C_{pre}. Here are some definitions:

C_{save} is the part where the data traffic of single-round training of the proposed algorithm is less than that of IABFL, which is shown as

$$\begin{aligned} C_{\text{save}} = C_{\text{pre}} - C_{\text{mod}} &= n^2 t - nt + n^2 - n - bnt \\ &- b^2 t + bt + n - b^2 + b. \end{aligned} \tag{9}$$

R_{save} is the saving coefficient, that is, the ratio of the saved data traffic to the traffic consumed when transmitting a single model:

$$R_{\text{save}} = \frac{C_{\text{save}}}{t} \cong n^2 - b^2 + (1 - n)b - 2n. \tag{10}$$

Because the saving factor only indicates that the data traffic of a single-round training of the improved model is better than that of the original model if it is greater than 0, so $R_{\text{save}} > 0$ and the minimum number of consensus nodes g equals two; this leads to the following equation:

$$g \in \left[2, \frac{\sqrt{5n^2 - 10n + 1} - n + 1}{2}\right), \tag{11}$$

where n is an integer $(n \geq 5)$, so

$$\frac{\sqrt{5n^2 - 10n + 1} - n + 1}{2} \geq 2. \tag{12}$$

According to the calculation, the algorithm has an impact only when the number of participants in the federated learning is greater than or equal to five. The next derivation is discussed based on this result.

The security value of the improved model, C_{safe}, has two components: (i) the security of the blockchain and (ii) the security of the edge node. The principle of the blockchain states that the more the consensus nodes, the stronger the various attacks and the more secure the entire blockchain. The security value of the blockchain is denoted as g. There are more edge node workers in a single cluster for larger n/g values, which would "dilute" the polymerization influence degree of a single node of the local model. Besides, the safety value of the model is negatively correlated with n/g. Hence, g/n is taken as the security value of the edge node, which can be calculated as

$$C_{\text{safe}} = (s + 1) \times \left(g^m + (g/n)\right), \tag{13}$$

where $s \in [0, 10]$ is the security value.

The safety factor of the model proposed in this article is denoted as R_{safe}. According to the principle of blockchain, the more consensus nodes, the stronger the ability to resist various attacks, such as the Byzantine [24], and the more secure the whole blockchain; therefore, the more consensus nodes the model has, the lower the proportion of security value will be got. Hence, R_{safe} is negatively correlated with the size of g; that is,

$$R_{\text{safe}} = \frac{C_{\text{save}}}{g} \cong (s + 1) \times g^{m-1}. \tag{14}$$

Next, the value of m and its optimal value in the actual application are analyzed by modeling. The overall equalization coefficient of this model ϑ is defined as follows:

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}}. \tag{15}$$

For $m = 2$, ϑ can be calculated as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - 2n}{(s + 1)g} - \frac{g}{s + 1} + \frac{1 - n}{s + 1}. \tag{16}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 4 and 5, respectively.

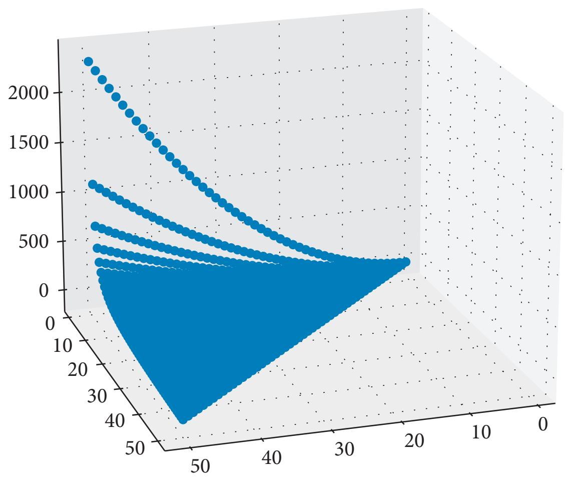


Figure 4: Value of ϑ for n and g, when $m = 2$.

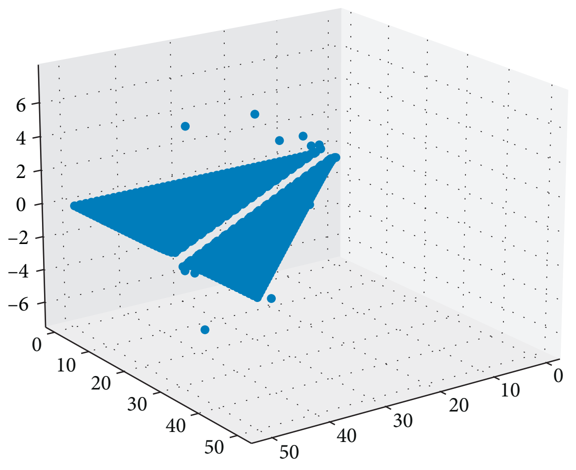


Figure 5: Value of $1/\vartheta$ for n and g, when $m = 2$.

When $m = 3$, ϑ is calculated as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - g^2 + (1-n)g - 2n}{(s+1)g^2}. \tag{17}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 6 and 7, respectively.

When $m = 4$, $1/\vartheta$ is calculated as

$$\frac{1}{\vartheta} = \frac{R_{\text{safe}}}{R_{\text{save}}} = \frac{(s+1)g^3}{n^2 - g^2 + (1-n)g - 2n}. \tag{18}$$

When s is constant, the values of ϑ and $1/\vartheta$ are shown in Figures 8 and 9, respectively.

When $m = 2$, Figures 4 and 5 show that the proportion of R_{save} in the balance coefficient is super linear growth with n, which does not meet the actual situation and expectations. When $m = 4$, Figures 8 and 9 show that the proportion of R_{save} in the balance coefficient is super linear growth with g. Similarly, when $m \geq 4$, it is more incompatible. When $m = 3$, Figures 6 and 7 show that the balance coefficient trend is relatively stable with the relationship between n and g, which conforms to the actual situation and expectation and then further verification.

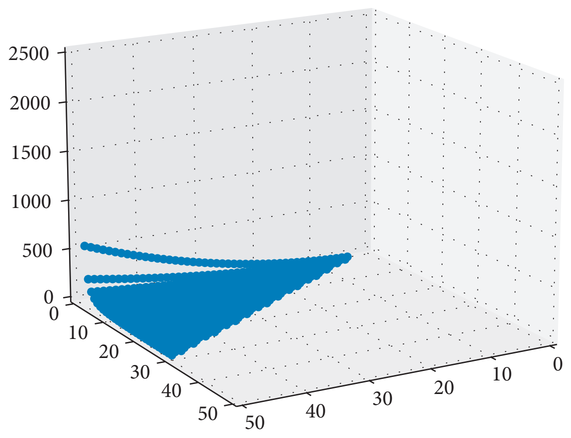


Figure 6: Value of ϑ for n and g, when $m = 3$.

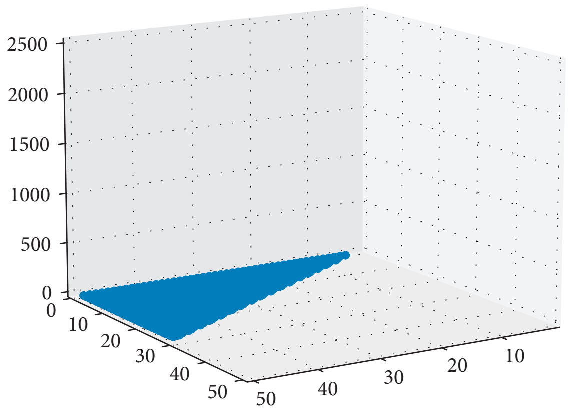


Figure 7: Value of $1/\vartheta$ for n and g, when $m = 3$.

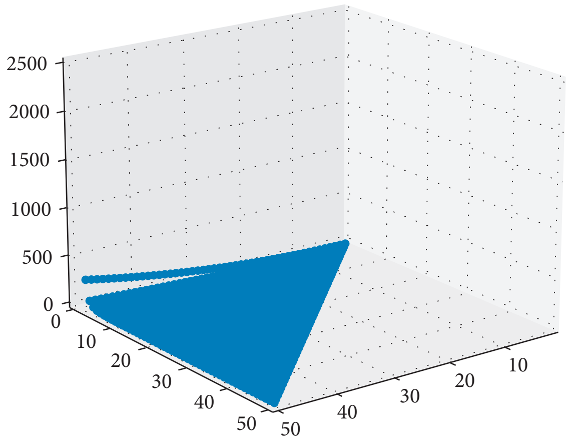


Figure 8: Value of ϑ for n and g, when $m = 4$.

When $m = 3$, the number of the edge nodes in a single cluster, n/g, is taken as a variable, and its relationship with the balance coefficient is shown in Figure 10.

Figure 10 shows that the proportion of R_{save} in the balance coefficient increases linearly with the number of edge nodes in a single cluster, n/g. When $g > 5$, the impact of g on the overall trend is almost invisible. Also, it is intuitively clear from the formula that the equilibrium factor ϑ is

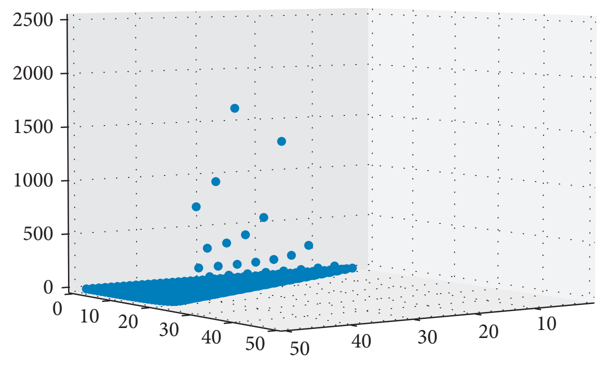


Figure 9: Value of $1/\vartheta$ for n and g, when $m = 4$.

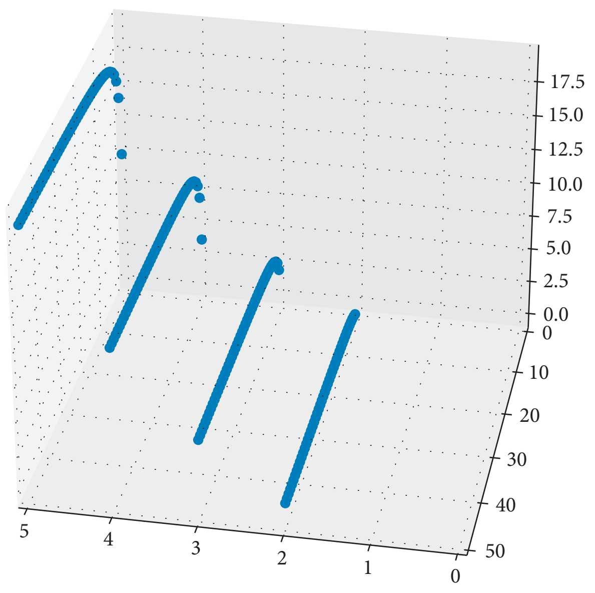


Figure 10: Value of ϑ for n/g and g when $m = 3$.

inversely proportional to the safety value s. It is concluded that when $m = 3$, the balance coefficient ϑ complies with the actual situation and the expectation of the model. Thus, the balance coefficient ϑ is determined as

$$\vartheta = \frac{R_{\text{save}}}{R_{\text{safe}}} = \frac{n^2 - g^2 + (1-n)g - 2n}{(s+1)g^2}. \tag{19}$$

As defined, when $\vartheta = 1$. The model focuses on saving the communication cost of the whole federated learning architecture when $\vartheta = 1$, the model is in a relative balance between saving communication consumption and model security. The model focuses on the safety of the entire federated learning architecture when $\vartheta < 1$. The specific value can be personalized by the requester during the training task, and this paper is recommended to take the balance state $\vartheta = 1$.

When $\vartheta = 1$, $R_{\text{save}} = R_{\text{safe}}$ shown as follows:

$$R_{\text{save}} - R_{\text{safe}} \cong n^2 - (s+2) \times g^2 - n \times g = 0, \tag{20}$$

where g is

$$g = \frac{\sqrt{4 \times s + 9} - 1}{2 \times s + 4} \times n. \tag{21}$$

Using (21), the relational Table 3 is generated.

The table shows that even if the value of security is minimized, it can save approximately 25% of the communication costs. When the security value $s \geq 2.79$, it can save about 50% of the communication costs.

3.4. *Feasibility Analysis of Double-Layer Aggregation Model.*

The research indicates that the core step in the FedAvg algorithm is the parameter aggregation of each submodel, which can be formulated as

$$\omega_{t+1} \longleftarrow \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}. \tag{22}$$

In (22), the global parameter is obtained by accumulating k subparameters based on the weight of the sample data size. The total information of each subparameter consists of two parts: (i) ω_{t+1}^{k}, the value of the submodel's current parameter, and (ii) n_k, the number of samples trained with the model parameter (in federated learning, it often refers to an edge node participating in training the amount of data). From another perspective, if a subparameter contains the value of the submodel's current parameter and the number of samples from which the model parameter is trained, then the subparameter can be regarded as produced by the same training sample.

Therefore, according to the combined rate of addition, it can be initially obtained that the result of the global parameter can be obtained by weighting the parameters of the two submodels, as shown in the following equation:

$$\omega \longleftarrow \frac{n_1}{n_1 + n_2} \omega^1 + \frac{n_2}{n_1 + n_2} \omega^2. \tag{23}$$

Now, let us expand the added submodel into i $(1 < i < k)$ parts. First, the k_i training samples are aggregated according to FedAvg to obtain the global parameter ω_i as

$$\omega \longleftarrow \sum_{k=1}^{K} \frac{n_k}{n} \omega_k, \tag{24}$$

where n_k are samples contained in each training sample. Then, the k training samples are randomly divided into i $(1 < i < k)$ parts. The i part has a total of k_i training samples, and each training sample is aggregated according to FedAvg to obtain the local parameter ω_i. At this time, ω_i is equivalent to a training sample of $i \times n_k$ samples. Then, i training samples ω_i are aggregated according to FedAvg to obtain the global parameter ω_α as follows:

$$\omega_\alpha \longleftarrow \sum_{i=1}^{I} \frac{n_i}{n_\alpha} \omega_i, \tag{25}$$

where n_α is

$$n_\alpha = \sum_{i=1}^{I} n_i. \tag{26}$$

Table 3: Communication volume reduction ratio table.

Value of security value, s	The average number of edge nodes in each cluster, n/g	The proportion of reduction in communication volume (%), $C_{\text{save}}/C_{\text{pre}}$
0	2.00	25
1	2.30	37
2	2.56	45
3	2.79	51
4	3.00	55
5	3.19	59
6	3.37	62
7	3.54	64
8	3.70	66
9	3.85	67
10	4.00	68

Finally, the following equation is gathered:

$$\begin{aligned}\omega &\longleftarrow \sum_{k=1}^{K} \frac{n_\alpha}{n}\omega_\alpha \\ &\longleftarrow \sum_{k=1}^{K} \frac{n_\alpha}{n} \times \frac{n_1 \times \omega_1 + \cdots + n_i \times \omega_i}{n_\alpha} \\ &\longleftarrow \sum_{k=1}^{K} \frac{n_k}{n}\omega_k.\end{aligned} \tag{27}$$

The final results of ω and ω_α demonstrate that they are equivalent. Thus, it can be concluded that the double-layer aggregation does not affect the final result of the global model parameters. The same applies to the IABFL, where the following equation expresses the average aggregation based on the number of nodes:

$$\omega_{i,e}' \longleftarrow \frac{1}{k} \sum_{m \in M_{i,e}} \omega_{m,e-1}. \tag{28}$$

3.5. Experiments and Result Analysis

3.5.1. Experiments. The federated learning process of the IABFL and the double-layer aggregation model is simulated through multiple virtual nodes. The aggregation effect, aggregation speed, and the training communication volume between the two models are compared. The results revealed the optimization effect in the availability and communication cost of the double-layer aggregation model.

The experiments adopt the controlling variables method. Experiment 1 is to compare the accuracy of the two models, while Experiment 2 compares them in terms of aggregation time. Experiment 3 evaluates the required network/data traffic, that is, communication cost, when the two models are aggregated. The experiments use the MINIST training set as the local data set, and 20% of the same validation set is used as the test set.

In the experiments, we simulated a small-scale federated learning environment, where the number of participating nodes is set to 50, the data set size of each node to 1000, and the number of local iterations of single-round training to 10.

Table 4: Experimental testbed specifications.

Environment	Model/version
System	64-bit Windows 10 Professional operating system
CPU	Intel® Core™ i7-7700 HQ@2.80GHz
GPU	NVIDIA GeForce GTX 1050
Software	Anaconda3
Python	Python = 3.7
Dependency	PyTorch, PySyft

As mentioned, each iteration used 20% of the local data. We recorded the accuracy, aggregation time, and network traffic after each round of the training and averaged the values after ten times of running.

The experimental environment is given in Table 4.

3.5.2. Analysis of the Results

Experiment 1. Comparison of the aggregation effects

The two aggregation models are tested under the same conditions. Figure 11 illustrates that although the accuracy of the IABFL in the first three rounds is not as good as the double-layer aggregation, it becomes slightly better in the following rounds. Hence, the final difference in accuracy between the two models is very small. Considering the amount of data used, the accuracy that the models achieved is about 84%, which is quite good. This experiment validates that the double-layer aggregation model using the safety evaluation method proposed in this paper is applicable.

Experiment 2. Comparison of the polymerization speeds

Figure 12 depicts that the double-layer aggregation model consumes more time than the IABFL. According to the seven rounds repeated ten times, the average time of each round increases by 0.206 seconds. Analyzing the system process shows that the increased time is mainly used for the local aggregation of consensus nodes and the time consumed by one more segment of network communication. However, the increased time only accounts for 2.53% of the total model training, and it does not affect the training timeliness. This experiment, therefore, validates that the polymerization aging of the double-layer aggregation model proposed in this paper is applicable.

Experiment 3. Comparison of communication costs

Figure 13 shows that the communication cost of the double-layer aggregation model gradually decreases with the increase of the safety value. Hence, the security value s should be set in [0, 4] if there is no special requirement. It can also be seen that although the traffic trend of the double-layer aggregation model is the same as the previous theoretical analysis, there is always a small gap between the two, which is constant. This difference might be due to several factors, such as the network protocol header, various verification packets (e.g., the three-way handshake packet), and network fluctuations. However, the model is very close to the theoretical value, and the difference does not affect the actual use. Therefore, the model proposed in this paper can still reduce a large amount of data traffic and hence the

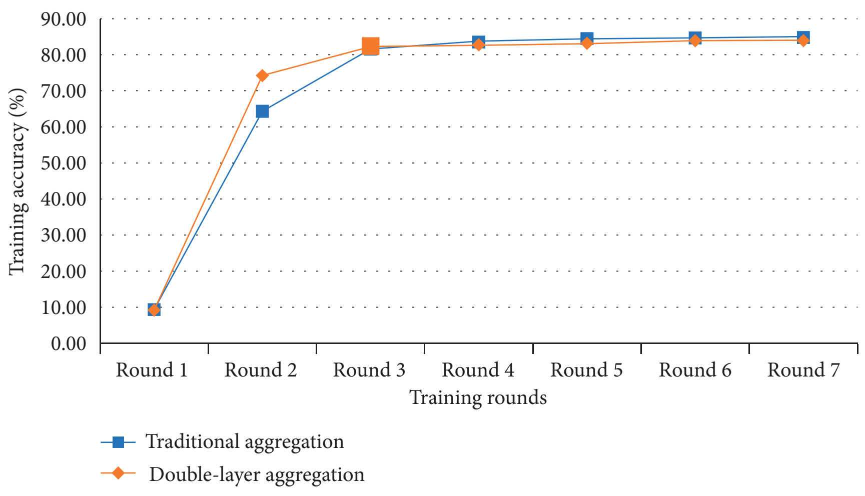


FIGURE 11: Training round versus accuracy of the two aggregation models.

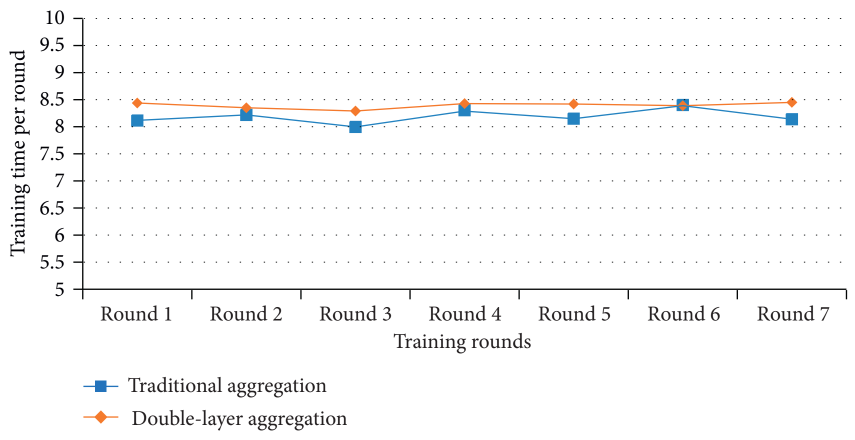


FIGURE 12: Training round versus time of the two aggregation models.

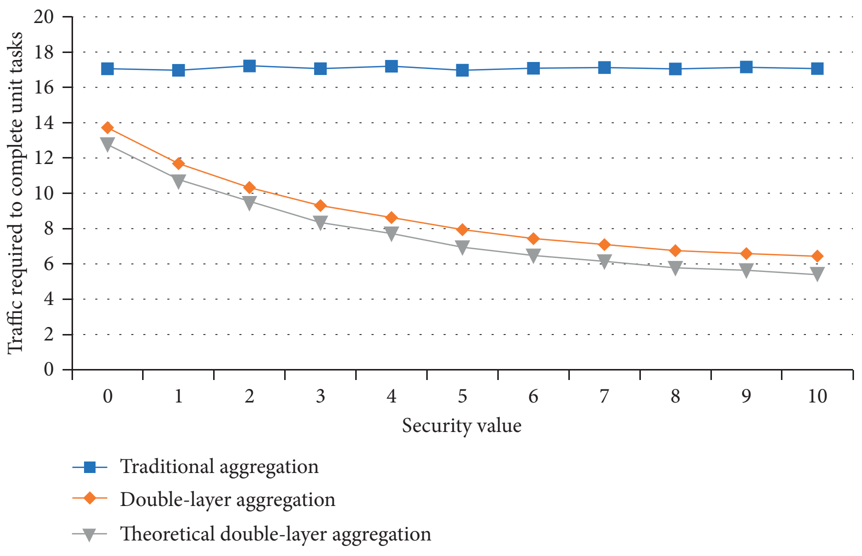


FIGURE 13: Security level versus communication volume.

communication cost during model training, which meets the design goals.

The above analysis of the experimental results concludes that the blockchain-based communication cost optimization and safety evaluation methods proposed can reduce the communication cost of the learning process while ensuring the security and availability of the nodes.

4. Conclusions

For the existing protection methods of "free-riding attacks" and "model poisoning attack" in the federated learning process, the training models need to be audited one by one, causing the problem of high communication costs throughout the training process. A step-by-step aggregation federated learning platform based on blockchain is proposed, and the architecture and algorithm of the platform are designed in detail. In this article, we first studied the verification mechanism of useless or malicious nodes. Then, using a competitive voting verification algorithm, a blockchain federation learning communication cost optimization method based on security evaluation is proposed, and the security of the model is analyzed in combination with game theory methods. Finally, through experimental comparison and analysis, we verify that the proposed method can reduce the communication cost of the learning process while ensuring the security and availability of IoT nodes.

Data Availability

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

Conflicts of Interest

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

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant nos. 61802086 and U20B2048), Defence Industrial Technology Development Program (grant no. JCKY2020604B004), and the Fundamental Research Funds for the Central Universities (grant nos. 3072021CF0608 and 3072020CF0601).

References

[1] H. Sun, Z. Liu, G. Wang, W. Lian, and J. Ma, "Intelligent analysis of medical big data based on deep learning," *IEEE Access*, vol. 7, pp. 142022–142037, 2019.

[2] S. Santiago, G. Boris, R. Eduardo, M. T. Paul, A. Andre, and L. Marco, "Federated learning in distributed medical databases: meta-analysis of large-scale subcortical brain data," in *Proceedings of the 16th IEEE International Symposium on Biomedical Imaging*, pp. 270–274, IEEE, Venice, Italy, April 2019.

[3] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: a cloud-edge based framework," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 35–44, 2020.

[4] A.-M Fu, X.-L. Zhang, N.-X. Xiong, Y. Gao, H. Wang, and J. Zhang, "VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT," *IEEE Transactions on Industrial Informatics*, 2020.

[5] H. Kim, P. J. Hong, B. Mehdi, and K. Seong-Lyun, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[6] X.-L. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "FLChain: a blockchain for auditable federated learning with trust and incentive," in *Proceedings of the 5th International Conference on Big Data Computing and Communications (Bigcom)*, pp. 151–159, IEEE, Qing Dao, China, August 2019.

[7] U. Majeed and H. C. Seon, "FLchain: federated learning via MEC-enabled blockchain network," in *Proceedings of the 20th Asia-Pacific Network Operations And Management Symposium (APNOMS)*, pp. 1–4, IEEE, Matsue, Japan, September 2019.

[8] W. Zhang, Q. Wang, and M. Li, "Medical image collaborative training based on multi-blockchain," in *Proceedings 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 590–597, IEEE, San Diego, CA, USA, November 2019.

[9] B. Yin, H. Yin, Y. Wu, and Z. Jiang, "FDC: a secure federated deep learning mechanism for data collaborations in the internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6348–6359, 2020.

[10] Y. J. Kim and C. S. Hong, "Blockchain-based node-aware dynamic weighting methods for improving federated learning performance," in *Proceedings of the 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 1–4, IEEE, Matsue, Japan, November 2019.

[11] I. Martinez, S. Francis, and A. S. Hafid, "Record and reward federated learning contributions with blockchain," in *Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 50–57, IEEE, Guilin, China, October 2019.

[12] W. Zhang, Q. Lu, Q. Yu et al., "Blockchain-based federated learning for device failure detection in industrial IoT," *IEEE Internet of Things Journal*, vol. 8, 2020.

[13] Y.-L. Lu, X.-H. Huang, Y.-Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2020.

[14] Y. Zhao, J. Zhao, L. Jiang et al., "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[15] C. Yu, L. Fang, L. Tong, X. Tao, L. Zheli, and L. Jin, "A training-integrity privacy-preserving federated learning scheme with trusted execution environment—science direct," *Information Sciences*, vol. 522, pp. 69–79, 2020.

[16] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[17] X. Wang, S. Garg, H. Lin, G. Kaddoum, J. Hu, and M. S. Hossain, "A secure data aggregation strategy in edge computing and blockchain empowered Internet of things," *IEEE Internet of Things Journal*, vol. 99, p. 1, 2020.

[18] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, pp. 395–403, Los Angeles, CA, USA, December 2019.

[19] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-Enabled smart contracts: architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.

[20] Y. Huang, B. Wang, and Y. Wang, "MResearch on Ethereum private blockchain multi-nodes platform," in *Proceedings of the 2020 International Conference on Big Data, artificial intelligence and Internet of things engineering (ICBAIE)*, pp. 369–372, IEEE, Fuzhou, China, June 2020.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, IEEE, Santiago, Chile, December 2015.

[22] Y. Lu, Q. Tang, and G. Wang, "On enabling machine learning tasks atop public blockchains: a crowdsourcing approach," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 81–88, IEEE, Singapore, November 2018.

[23] D. Pietro, K. A. Ellersgaard, S. Čedomir, and P. Petar, "Analysis of the communication traffic for blockchain synchronization of IoT devices," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–7, IEEE, Kansas, MO, USA, May 2018.

[24] R. Kashyap, K. Aror, M. Sharma, and A. Aazam, "Security-Aware ga based practical byzantine fault tolerance for permissioned blockchain," in *Proceedings of the 4th International Conference on Control, Robotics and Cybernetics*, pp. 162–168, IEEE, Tokyo, Japan, September 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5561816, 23 pages
https://doi.org/10.1155/2021/5561816

Review Article

Comparing and Analyzing Applications of Intelligent Techniques in Cyberattack Detection

Priyanka Dixit,[1] **Rashi Kohli**,[2] **Angel Acevedo-Duque**,[3] **Romel Ramon Gonzalez-Diaz**,[4] **and Rutvij H. Jhaveri**[5]

[1]*RGPV University, Bhopal, Madhya Pradesh, India*
[2]*IEEE, New York, NY, USA*
[3]*Faculty of Business and Administration, Public Policy Observatory, Universidad Autónoma de Chile, Santiago, Chile*
[4]*Centro Internacional de Investigacióny Desarrollo (CIID), Monteria 230001, Colombia*
[5]*Pandit Deendayal Energy University, Gandhinagar, Gujarat, India*

Correspondence should be addressed to Angel Acevedo-Duque; angel.acevedo@uautonoma.cl

Received 21 February 2021; Revised 7 May 2021; Accepted 22 May 2021; Published 14 June 2021

Academic Editor: Muhammad Shafiq

Now a day's advancement in technology increases the use of automation, mobility, smart devices, and application over the Internet that can create serious problems for protection and the privacy of digital data and raised the global security issues. Therefore, the necessity of intelligent systems or techniques can prevent and protect the data over the network. Cyberattack is the most prominent problem of cybersecurity and now a challenging area of research for scientists and researchers. These attacks may destroy data, system, and resources and sometimes may damage the whole network. Previously numerous traditional techniques were used for the detection and mitigation of cyberattack, but the techniques are not efficient for new attacks. Today's machine learning and metaheuristic techniques are popularly applied in different areas to achieve efficient computation and fast processing of complex data of the network. This paper is discussing the improvements and enhancement of security models, frameworks for the detection of cyberattacks, and prevention by using different machine learning and optimization techniques in the domain of cybersecurity. This paper is focused on the literature of different metaheuristic algorithms for optimal feature selection and machine learning techniques for the classification of attacks, and some of the prominent algorithms such as GA, evolutionary, PSO, machine learning, and others are discussed in detail. This study provides descriptions and tutorials that can be referred from various literature citations, references, or latest research papers. The techniques discussed are efficiently applied with high performance for detection, mitigation, and identification of cyberattacks and provide a security mechanism over the network. Hence, this survey presents the description of various existing intelligent techniques, attack datasets, different observations, and comparative studies in detail.

1. Introduction

An excessive use of the Internet in various areas are encouraging the researchers and scientists to use intelligent systems that can support the users, and different applications also ensure efficient computation, maintaining the quality of service over the network. The traditional methods were time-consuming, less efficient, and giving average performance and cannot fit for providing the solution for complex, multiobjective, or real-world problems. Hence, the necessity of efficient attack detection systems can reduce the harmful effects of cyber threats [1]. Cybersecurity is the collection of various technologies and security mechanisms that develop for the protection of data, information, network, a program from the different attack activities such as data modification, stealing, unauthorized access, and destruction over the Internet or network. Cybersecurity components concern mainly host protection and network security systems [2]. Currently it is used to protect many areas such as cloud computing [3], wireless sensor network [4], and IoT. There are a lot of security measures which are available for providing security to the systems or networks such as antivirus, firewall, and IDS. However, still, cyber threats continuously harm and disrupt Internet services every day. This motivates many researchers for providing their extensive contribution to design security systems [5–9].

The following are the popular cyberattacks such as denial of service attack, distributed denial of service attack [10], remote to local attack, probing, user to root attack, adversarial attacks, poisoning and evasion attacks [11], botnet [7], phishing attack [12], spamming [13], and zero-day attack [6].

Many different methods are used for attack detection that broadly categorized into three major categories such as anomaly-, misuse-, and hybrid-based detection. Misuse-based detection can be scanned by prestored attack signatures and mostly used to detect identified attacks. It is useful to detect known attacks with minimum false alarms. It requires a certain modification of the signature and rules of attacks on the database.

The anomaly-based technique is capable to detect both the attack types either known or unknown. It can capture network and host machine behavior and also determines anomalies as deriving from normal behavior. It is the most popular method because it can detect zero-day attacks. There are many merits of using this method, and one of them is the customization of profiling actions due to which attackers get confused about which activity they follow to enter and remain undetected. However, besides the merits, there is a drawback also it evaluates with very high false alarm rates and sometimes the legitimate activity considered as an anomaly.

Another is a hybrid technique, a fusion of anomaly- and misuse-based detection. It supports high performance in the detection phase and a minimum false alarm rate.

Here presents some existing research that is lighting the contribution of machine learning and metaheuristic techniques in cyberattack detection, especially focusing on better classification and optimal feature extraction also with their results. Tu et al. [14] proposed hybridization of PSO and SVM for feature extraction, and in this method, fitness function of PSO is used for classification.

Athari and Borna [15] proposed a hybrid metaheuristic particle swarm intelligence, genetic algorithm (GA), and glowworm which are collectively used for classification and optimal feature extraction in the wireless sensor network. The purpose of using the metaheuristic algorithm is to solve the problems such as low convergence and low local optimality. In this paper, certain parameters were calculated as permittivity against DoS attack, reliability, number of active nodes, and energy consumption. The results shown by PSO have the highest permittivity, reliability, and larger number of active nodes compared to the genetic algorithm and glowworm optimization technique for DoS attack, GA has less permittivity, reliability, and number of active nodes than PSO and GSO, and energy consumption of GA is very low compared with the above two techniques because of its simplicity. The bioinspired algorithms are popularly used for the optimal feature selection and solving optimization problems in different fields compared with data mining techniques that were previously used in classification and feature selection in different applications such as pattern recognition, intrusion detection, clustering, and data classification.

Sagarin and Taylor [16] proposed a biological evolutionary system for providing better approaches in the field of security. Jamali and Shaker [17] proposed a metaheuristic approach for recognizing denial of service attack type on TCP protocol called TCP SYN flood attack that requests for TCP connections in the form of huge flood request to the server. This attack detection framework is designed by particle swarm optimization (PSO) and the queuing model for optimally using the buffer space and solving attack recognition problem over the network.

Tarao and Okamato [18] used an artificial immune algorithm of the metaheuristic family to model the framework for DoS attack detection to overcome the vulnerabilities of the server-side. By this technique, the false alarm rate is minimized and detection performance is simulated by the machine learning approach. Metaheuristic algorithms are very efficiently used in cybersecurity for the implementation of the attack recognition framework with high learning capabilities. Bhattacharya et al. [19] proposed a hybrid principal component analysis and firefly-based model to classify intrusion detection system datasets. The model performs one-hot encoding for the transformation of the attack datasets and then hybrid PCA-firefly algorithm used for dimensional reduction. The another XGBoost algorithm is used for classification of attacks. This hybrid model perform well by achieving high accuracy of 99.9, sensitivity 93.1, and specificity 99.9.

Visumathi and Shunmuganathan [20] proposed intelligent computational techniques such as SOM, SVM, multilayer perceptron (MLP), Bayesian network (BN), and logistic regression for the classification of attack data. Srinoy [21] proposed a hybrid combination of particle swarm intelligence for an optimized feature selection and support vector machine (SVM) that classify attack data. After the result is evaluated, it had been found that that the above-mentioned hybrid technique can easily identify not only known attacks but also detect the early apprehensive activities that cause unknown attack. This method is efficiently solved feature selection problem and achieved detection rate of 96.11% with high classification accuracy.

Mourougan and Aramudhan [22] proposed a computational model for solving classification problems and extracting features by the hybrid combination of the PSO technique and the GA algorithm. The proposed model of attack detection can identify DoS attack with maximum detection accuracy and minimum false alarms by genetic particle swarm intelligence-based binding feature extraction that is mostly used for intrusion feature selection. Results are shown with maximum accuracy and minimum false alarms as compared with the fuzzy clustering technique.

Akyazi and Sima Uyar [23] proposed the model which was built on an anomaly-based intrusion detection method and using the artificial immune system (AIS) to improve multiobjective evolutionary algorithm, to get the better performance of the proposed model for the detection of DDoS attack tested on the DARPA-based LLDOS 1.0 dataset. The proposed model is applied iteratively for computation, and if we find the negative selection, then we redefine the objectives with the same concept. The zero-percent false positive rate is found by applying the above approach, and hence, results show that the method is successful with better accuracy.

Ben Sujitha and Kavitha [24] proposed a method that can efficiently detect cyberattack and provide better accuracy and efficiency. For performance improvement in the attack detection model, optimized features selection approach was used. The proposed system is built to provide an optimal feature extraction algorithm to construct summarized features applied to the multiobjective PSO algorithm. The anomaly detection method was applied, and the proposed system was tested on the KDDCUP99 intrusion dataset. The result of the proposed system shows that it can successfully deal with real-time attacks worked with high speed.

The rest of the paper is arranged as follows: Section 2 focuses on important steps of classification and machine learning. Section 3 focuses on the detail of different metaheuristic algorithms used in cyberattack detection. Section 4 describes the different machine learning techniques used in cyberattack detection. Section 5 focuses on different datasets. Section 6 presents observations and evaluations. Section 7 presents challenges and future directions, and finally, Section 8 concludes the paper.

2. Important Steps of Classification and Machine Learning

Many techniques were previously used in knowledge discovery of database (KDD), especially data mining techniques such as clustering and data classification techniques. KDD is dealing with extracting useful information from the data source. In Figure 1, the various steps for extracting knowledge are data preparation, data selection, data cleaning, and extracting features or patterns from the data. According to Periyar and Salem [22], data preprocessing is a most essential step of machine learning computation that can remove noisy data such as repeated values, out-of-limit values, irrelevant data logics, checking null values, and missing terms or instances. The data preprocessing has certain steps such as learning, normalization, transformation, feature selection, and extraction. Outcomes of preprocessed data are input or works as the training sets to extract knowledge for the testing phase. The precision of any classifier depends on selecting the optimal feature upsets from the original data [22].

Feature selection is the most essential step of data preprocessing, used before the classification process [24]. This method is useful to reduction of some repeated data patterns and noisy and unnecessary features, which is very useful to achieve accuracy in classification and improves attack detection rate. It is the method of selecting some subset of the actual features and can generate different new features [24]. FS has to perform two basic objectives firstly to provide accuracy in classification performance and reduce the number of features. Complex datasets sometimes degrade classification performance in the attack detection process, and it can create problems such as irrelevant data and repeated features, uncertainty, and ambiguity. These certain problems are obstacles not only in concern of

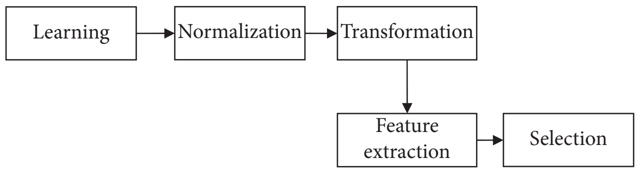


Figure 1: Steps of preprocessing of data.

detection speed but also in the performance of the detection process [24].

The approach comprises two major phases which are training phase and testing phase, and these phases are processed by using the following steps:

(1) Identification of different features, attributes, or classes of data during the phase of reprocessing these attributes which are extracted from the data

(2) Selection of attributes that is useful for the classification

(3) Learning processed by the help of training data

(4) Training the model used for the detection of unknown threats

These abovementioned are the various steps that are followed to process machine learning. In the training phase, signature-based classes are learned by using some training sets. In the testing phase, testing of new data is carried out by the classifier and they are checked whether they match with that class or not.

In another anomaly-based approach, the regular traffic data are defined in the training phase, this trained model is applied to the new data in the testing phase, and finally, testing sets are classified as a normal or malicious one.

In many research papers, machine learning (ML) is broadly categorised into three phases such as the first training phase, secondly testing phase, and the last validation phase. Machine learning has numerous methods for the training and testing process, some of the popular methods are artificial neural network (ANN) methods such as SVM, SOM, and multiple layer perceptron network (MLP), and these techniques have different parameters such as the number of layers, nodes, and processing units. When the training phase is completed, then a number of models are available and the selection of the model depends on its efficiency, accuracy, and error estimation. .

There are the following three types of ML methods which are broadly classified as supervised, unsupervised, and semisupervised [2]. When the model is trained by certain rules (training sets) and the data are well labeled, then it comes under supervised learning. Most of the supervised anomaly techniques were proposed using a support vector machine (SVM), multilevel perceptron network (MLP), and decision tree [25].

When some part of the dataset is labeled by preprocessing of data methods due to which the problems introduce, that comes under semisupervised learning. If the dataset is unlabeled, then some problems arise in extracting various attributes, classes, structures, and patterns from those data, and such problems come under unsupervised learning [2]. Which is the model or leaning method used depends on the problem that should be solved. Hence, according to the problem, the best-suited learning approach is used.

Once the steps of the classification model are completed such as training, validation, and testing sets, hence the model is able to be preferred in the future for further problem-solving strategies. There are many machine learning methods which are available for solving any classification problem efficiently such as artificial neural network learning methods, both supervised or unsupervised learning techniques such as self-organizing map, linear logistic regression, and other feedforward neural network methods, naïve Bayes, support vector machine (SVM), and multilevel perceptron (MLP) classifiers. These different methods were applied to different benchmarks and popularly used for solving classification problems such as well-known KDDCUP99 dataset for intrusion detection according to Srinoy et al. [21] using the anomaly-based approach with a hybrid form of PSO and SVM for the optimal feature selection and classification tasks.

According to Shinde and Parvat [26], using the NSL-KDD dataset, we apply the hybrid form of PSO and ABC on SVM for solving feature selection and classification problems to achieve high DR and low FAR. Prasad et al. [27] analyzed metaheuristic anomaly-based algorithms for real-time detection of application layer distributed denial of service attack successfully detected by using the hybrid combination of cuckoo search, bat, and firefly algorithm and proved to be an efficient technique by improving the parameters such as accuracy, efficiency, and performance analysis. Jadidi et al. [28] proposed multilevel perceptron (MLP) based on the anomaly attack detection method in a high-speed network. The PSOGSA and cuckoo algorithms based hybrid approach was used that ensures improved accuracy to classify abnormal traffic.

Akyazi and Sima Uyar [23] proposed a model for attack detection against the DoS attack by using the AIS algorithm based on anomaly detection and applied on the DARPA LLDOS 1.0 dataset that provides an efficient result, high TPR, and very low FPR. Hence, a multiobjective evolutionary algorithm is used inspired by AIS that is proved to be very effective for DDoS attack detection. Hence, machine learning methods are very popularly used for cyberattack detection and proved to be very efficient on various benchmarks. Today, for the better computational result, hybrid metaheuristic algorithms and ML approaches are used for optimal feature extraction, and in many classification problems, they specially deal with complex datasets.

In the above section of the paper discussing machine learning approaches with some current research studies, now after preprocessing, feature extraction, and classification of the data model we talk about the computational matrices of classification. There are several classification matrices which are used for machine learning in the attack-detection process. These matrices are discussed below in this part. The evaluation can be done on four main parameters such as false positive (FP) attacks which are wrongly classified as attacks, true positive (TP) which shows that attacks are correctly classified, true negative (Tn) which shows that the system is correct in spotting normal conditions, and false negative (Fn) attacks which are correctly classified as attacks [29]. The following are the attack detection matrices based on the anomaly detection method:

(i) To measure the overall performance, four matrices are broadly used such as accuracy, error rate (ER), miss rate (MR), and false alarm rate (FAR) [28, 30].

(ii) Accuracy can be measured as

$A = (\mathrm{TP} + \mathrm{Tn})/(\mathrm{TP} + \mathrm{Tn} + \mathrm{FP} + \mathrm{Fn})$

Error rate, $\mathrm{ER} = (\mathrm{Fn} + \mathrm{FP})/(\mathrm{TP} + \mathrm{Tn} + \mathrm{FP} + \mathrm{Fn})$

Miss rate, $\mathrm{MR} = (\mathrm{Fn})/(\mathrm{TP} + \mathrm{Fn})$

False alarm rate, $\mathrm{FAR} = (\mathrm{FP})/(\mathrm{Tn} + \mathrm{Fp})$.

(iii) True negative rate, also called specificity, $\mathrm{TNR} = (\mathrm{Tn})/(\mathrm{Tn} + \mathrm{Tp})$ ratio of items which are correctly classified as negative [2].

(iv) True positive rate, also called as sensitivity or recall or detection rate, TPR $(\mathrm{Tp})/(\mathrm{Tp} + \mathrm{Fn})$ [2].

(v) Negative predictive value (NPV) ratio of items which are correctly classified as negative, $\mathrm{NPV} = (\mathrm{Tn})/(\mathrm{Tn} + \mathrm{Fn})$ [2].

(vi) FP rate or fall out rate ratio of items incorrectly classified as positive fall out $= (\mathrm{FP})/(\mathrm{Tn} + \mathrm{Fp})$ [2].

In the attack detection mechanism during the classification of data, certain metrics are evaluated as false alarm rate (FAR) and true positive rate (TPR). Both of the abovementioned matrices are directly proportional to relationships with each other. Both of FAR and TPR are plotted with the help of receiver-operating characteristics with different axes, x-axis for FAR and y-axis for TPR, when FAR increases, then TPR increases, and if FAS falls, the TPR falls [18]. The overall performance is measured by certain matrices such as total detection accuracy (TDA) that can be evaluated as a total sum of correctly classified data items to the sum of samples. The average detection time (ADT) is calculated as the total detection time to the total sum of samples [31], performance, class detection rate, detection rate, or false positive rate.

The performance matrices are also evaluated by recall or precision factors. Recall is measured as $\mathrm{TP}/\mathrm{TP} + \mathrm{Fn}$ and precision factor is measured as $\mathrm{TP}/(\mathrm{TP} + \mathrm{Fp})$, other matrices measured based on precision and recall are F-measures, and weight mean acts as a tradeoff between the above two. The F-measures was measured as $(2 * \mathrm{Recall} * \mathrm{Precision})/(\mathrm{Recall} + \mathrm{Precision})$ [32].

3. Metaheuristic Approaches for Optimal Feature Selection

In this section of the paper, we discuss various methods of metaheuristics that are broadly used in many areas for solving different complex optimization problems [5].

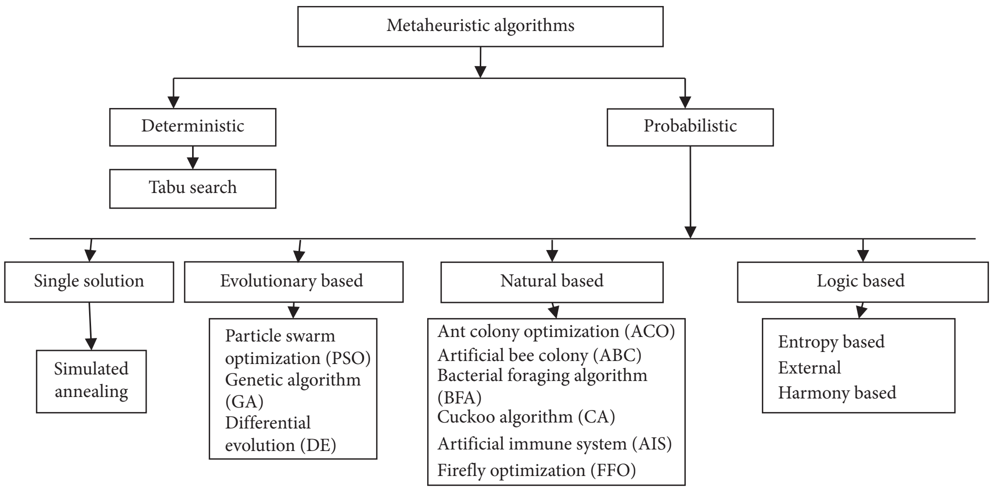


Figure 2: Categories of metaheuristic algorithms.

Figure 2 presents the classification of optimization such as swarm optimization techniques, genetic algorithm (GA), ant colony optimization (ACO), artificial immune algorithm (AIA), cuckoo algorithm (CA), artificial bee colony algorithm (ABC), and bacterial foraging algorithm (BFA), with their results by using the reference of different research papers and latest articles.

3.1. Particle Swarm Optimization. It is a widely used technique and one of the members of the swarm family which was firstly discussed by James Kennedy and Russell C. Eberhart in 1995 [14], and PSO is described in his first research paper "*A New Optimizer Using Particle Swarm Theory*." It is an intelligent optimization technique and a member of a class called metaheuristics. Particle swarm intelligence is stimulated by the socialized behavior of animals such as bird flocking and fish schooling nature.

Particle swarm intelligence is a simple yet powerful optimizing algorithm and also successfully applied to the number of applications of different areas and broadly in fields of science and engineering. According to its concept, each solution in the search space is considered a particle. Taking information from its surrounding particle will be in motion [15]. Its prototype can be implemented with ease programming and economical in terms of both storage and speed. Its computation and working steps are similar to the evolutionary and genetic algorithms [17]. PSO has the strong global best minima, each particle of the population has some randomized positions, and every particle is attached with some velocity. The velocity of the particle is adjusted through some previous behavior of each particle and its neighbors while roaming around the search space. All the particles update their positions and velocity to the best optimal value by communicating with each other [24].

Each particle in the problem space has certain coordinates that are connected along with some fitness value. This value is noted as P best. The other value also used by the PSO is considered as the best value that occurred up to any particle in the neighbors of the particle. Then, the particular location of the particle is noted as L best. As soon as a particle takes the population of its surrounding neighbors, the best value, called global best, is considered as best. Hence around this theory of particle swarm optimization every, time there is a change in velocity of each particle that value near its P best and L best locationsis selected [25, 33].

Let us consider that each particle is categorized into two parameters which are position vector and velocity vector, denoted as $(Y_i(t))$ and $(U_i(t))$. The location and velocity of particle i at the iteration of t times can be presented as

$$Y_i(t) = [Y_{i1}(t), Y_{i2}(t), \ldots\ldots, Y_{in}(t)], \tag{1}$$

$$U_i(t) = [U_{i1}(t), U_{i2}(t), \ldots\ldots, U_{in}(t)]. \tag{2}$$

Hence, the performance of the process, each particle having its independent knowledge P best, means that they have their own best value in the position, and collective knowledge G best means best of its best neighbor. The velocity will be updated using formula (3) [33–35]. The velocity of particles can be calculated as

$$\begin{aligned} U(\text{new}) &= w \times u_{\text{pd}}^{\text{old}} + C_1 \times \text{rand}_1 \times \text{pbest}_{\text{pd}} \\ &= y_{\text{pd}}^{\text{old}} + C_2 \times \text{rand} \times \text{gbest} - y^{\text{old}}, \end{aligned} \tag{3}$$

where w is inertia weight and is denoted as random number that may be considered between 0 and 1, and C1 and C2 are the constant value that can change the velocity of a particle towards the P best and G best, and its value can be set to 2 [25, 33, 34]. Hence, equation (4) represents update positions as

$$Y_i(t+1) = Y_i(t) + U_i(t+1). \tag{4}$$

Particle swarm optimization is applicable to various cyberattack detection systems for optimal feature selection problems and providing optimal solution. Here, we discuss some cyberattack detection systems designed in reference papers presented by various authors that use particle optimization for optimal results. Jamali and Shaker [17] proposed a detection model against denial of service attacks which are very prominent attacks over the Internet that block the legitimate users for gaining the network services. A defensive framework is proposed which had used particle swarm optimization to formulate various optimization problems to optimally solving the problem and improving the performance of an attack detection system with efficient consumption of buffer space.

In Hao et al.'s study [36] for achieving pattern recognition on weblog data for differentiating the normal and abnormal or malicious data, the user sessions are extracted from the log record. *k*-means clustering techniques are applied with a hybrid form of particle swarm optimization to generate an efficient attack detection model. This model successfully recognizes a DDoS attack with improved accuracy.

Momanyi Nyabuga et al. [37] provide defensive and preventive models against the denial of service (DoS) attack by applying the particle swarm intelligence algorithm in the VANET network. After the results, they found that PSO is efficient with high accuracy used for optimization in the attack detection system.

Shinde and Parvat [26] proposed a framework focused on achieving a high rate of attack detection with minimum false alarms by applying a hybrid form of SVM and swarm intelligence for selecting correct parameters. SVM had enabled us to provide an efficient classification of attack data. The attack detection framework uses knowledge gaining for selecting features and combines with support vector machine classifier. The important features will be selected by using an optimization approach called particle swarm intelligence and applied on the NSL-KDD dataset, and outcomes found high DR and low FAR after compared by regular SVM.

Guoli [38] presented the attack detection model based on which PSO is proposed with the Elman neural network. This fusion is used for optimal parameterization which improves performance. The experimented results are better as compared with traditional techniques.

3.2. Genetic Algorithm. **G**enetic algorithm (GA) is another metaheuristic technique that works on the concept of theory of evolution. The genetic algorithm (GA) is also called as population-based algorithms. These bioinspired algorithms are based on the iterative or repeating operations, and its basic ideology is adapted from genetics. Genetic algorithms can be designed as a simulation model in which the population of samples (chromosomes) from solution candidates in optimization problems will lead to an improved solution. One of the main features of the genetic algorithm is that it constantly works on chromosomes and solution space [15].

The population is mainly a collection of chromosomes in which each chromosome represents a certain position in the problem domain and probably a solution to the problem. By applying genetic operators on each population, they result in creating a new population that can have the same number of chromosomes. Then, a fitness function is determined for them, using operators are selection, crossover, and mutation, these are generally used by genetic algorithms, and a new generation can be created. The number of generations such as chromosome population is determined in the algorithm initialization step and methods of setting the parameter.

In the selection phase of GA, several chromosomes are selected from the existing chromosomes in a population to reproduction. Best chromosomes have more chances to be selected for reproduction. Chromosomes that make the next generation are being selected by this operator:

$$p_i = \frac{f_i}{\sum_{j=1}^{\text{pop-size}} f_j}, \tag{5}$$

$$l_i = \pi r^2 * p_i. \tag{6}$$

Equation (5) represents how to calculate the probability of each chromosome selection. p_i represents the probability of selection of ith chromosome, f_i is the fitness function value of the ith chromosome, and the dominator part of the equation shows the total amount of fitness function of all chromosomes. In equation (6), l_i is the length of the ith chromosome. Then, the crossover operator produces the child chromosome. This operator produces two-parent chromosome genes in the new chromosome (child). The chromosomes which were selected from the initial population as a parent for crossover operation are obtained from

$$n_c = 2\left[\frac{p_c \times n_{\text{pop}}}{2}\right]. \tag{7}$$

The mutation operator also selects a gene from the chromosome arbitrarily and then alters the content of that gene. The mutation operator guarantees that the genetic algorithm does not fall in the trap of local minimum point and covers all chromosomes which may be destroyed during the performance of other operations such as selection and crossover.

Genetic algorithms are based on the global search optima, and hence, they are efficiently used in the attack detection system; some of the following researchers use this technique for solving the optimization problem in attack data classification.

Siva Sankari et al. [39] proposed a model for the detection of a DoS attack also to observe the attacks over the Internet and predict the attack is DoS or not. In the proposed model, the genetic algorithm (GA) is used for optimizing features for optimal feature selection and identify DoS attacks. The GA is capable to learn the things itself and initiates the process of selection. It is used to generate optimal resolution for making the proper solution to complex problems. Genetic algorithms are implemented through the following steps such as selection, crossover, and mutation to find the optimal solutions. This approach is very accurate and efficient for identifying a DoS attack. The proposed system results showed better performance, and it is capable of detecting a DoS attack with high accuracy.

Mizukoshi and Munetomo [40] proposed the system which is designed for attack detection by learning the attack patterns and other anomalous traffic. The proposed system works as a real-time traffic pattern analyzer using GA for detecting abnormal traffic behavior. The system is built on using Hadoop distributed infrastructure, and the result shows the effectiveness of the DDoS defense system.

Lee et al. [41] proposed an approach which provides a defensive mechanism against DDoS attack by using a traffic matrix. In this work, they proposed an improved attack detection model that enhances the traffic matrix construction process and some particular parameters were optimized using the genetic algorithm (GA). The experiments were tested on DARPA 2000 and LBL-PKT-4 datasets, and the results were evaluated which provides better detection accuracy with a high speed as compared with previous techniques.

Bhuyan et al. [42] provide an inclusive survey on detection or prevention against DDoS attack and also its detection techniques with its tools in the different networks. The article also discussed the different issues, various challenges, and feasible solutions in the concerned domain.

In Dimitris et al.'s study [43], the neural network detector was designed against the detection of DDoS attack. For selecting optimal features, a genetic algorithm is used that can extract 44 statistical features from the packet header. The computation is based on a genetic algorithm that creates an error-free neural network-based DDoS detector. The experimental results have shown the improved succeed features for DDoS detection with high accuracy.

Lee et al. [44] proposed an attack detection model by improving some parameters of traffic matrix through GA to achieve optimization that utilizes a high attack detection rate. The traffic matrix construction operation improved by hash function for minimizing the rate of collisions also used the packet-based window size to minimize cost. The evaluation is applied on DARPA 2000 LLDOS 1.0 and LBL-PKT-4 attack datasets. The proposed work has shown high feasibility in concern of attack detection accuracy and speed.

3.3. Ant Colony Optimization Algorithm. It is a commonly used technique to resolve combinational optimization-based problems and belongs as a member of the metaheuristic family. This algorithm works as an agent-based system, simulates the behavior of ants to develop a learning-based system. The ants preferred to move in a straight line for food searching and protecting themselves from different situations, and firstly, they decide to move from left to right randomly. Then, some assumptions are taken such as the moving speed of each ant is the same and also depositing pheromone in the trail evenly. Hence, the ants prefer to move from left to right direction and will reach the food earlier, and pheromone accomplished the fast shortest path around the obstacles. While the other ants preferred to follow the way where they found the excess amount of chemical called pheromone, hence all the ants meet the target (source of food) through the shortest path.

The ant colony optimization is quite different from the traditional ant system in concern with the pheromone trails which can be updated in two phases. Firstly, when ants decide a tour, they can change the quantity of pheromone locally around routed boundaries by a local updation in position. Secondly, when each of the ants decides their tour, a global updation is applied to adjust the pheromone amount in the boundaries which is considered as best ant tour [21]. Hence, this phenomenon of optimization is used by different research studies for solving optimization problems. Along with the various applications of ant colony optimization, it is also used in cyberattack detection models successfully. Here, we discuss some of the research papers of its contribution in attack detection models. Dimitris et al. [43] proposed an ant colony system-based (DDIACS) framework for identification and detection of a low-rate distributed denial of service (LDDoS) attack detection, another well-known attack over the network. The proposed detection model is built with ant colony optimization, which is another strong optimization algorithm used to resolve complex optimization problems. The proposed framework has improved some parameters that are very complex while detecting multisource attacks such as flexibility, fast convergence, and robustness. This framework was tested upon the dataset DARPA and KDD. The outcomes have shown that the proposed method has successfully overcome the problem or errors with high accuracy than existing models. The proposed model found more than 89% of the detection rate and 83% accuracy.

Aldwairi et al. [45] proposed an anomaly-based detection model for the detection of unknown attacks. They proposed a model by using the ant colony optimization technique for selecting optimized features to improve the overall classification accuracy by rejecting unwanted features. In this proposed work, ant colony optimization of three levels of updating feature selection process had been proposed. This method efficiently used the information of each ant in the process of feature extraction and also improved the accuracy of the proposed system and classification of features. The evaluation results have shown that the proposed approach performed well as compared with previously used feature selection techniques.

3.4. Artificial Bee Colony. Swarm intelligence is a kind of self-organized system that can solve different optimization problems. Artificial bee colony (ABC) is another prominent optimization technique that works by the concept of imitating the foraging technique of bee swarms, firstly predicted by Visumathi and Shunmuganathan [20]. The artificial bee colony algorithm worked in the following three basic steps: the first is food source; it is based on some important factors such as the amount and quality of nectar, the total efforts for its extraction, and nearest to the colony. Secondly, foragers, employed foragers grasp information of food sources, and the third one is unemployed foragers continuously looking for food sources and broadly categorised into two types which are scout bees and onlooker bees. The whole process of searching food starts with scout bees sent for searching food sources in the colony in a random distribution manner. While the scout bees return, the food sources are rated by some threshold value and perform waggle dance [21]. The waggle dance is a unique interaction way and also helps to determine the food source direction through the nectar amount that represents fitness value. Onlooker bees choose the best food source by collecting all the information that is exposed by the waggle dances. This information helps them to reach the best sources without the help of any maps [46].

The following authors used artificial bee colony (ABC) in the attack-detecting system for solving optimization problems. Mahale and Gothawal [47] proposed an ABC algorithm to optimize some attributes of the artificial neural network, improve local optima problem, and also overcome low convergence speed of the neural network. The ABC algorithm can be efficiently used for finding the optimal solutions in minimum time. In this research work, the proposed algorithm was applied for attack detection and the evaluation outcome shows that the proposed method had performed and improved in some parameters such as DR and efficiency.

Priyadharshini and Kuppusamy [48] proposed an attack detection model based on anomaly detection techniques that detect attacks and improve performance by low false alarms. This proposed work is based on the ABC algorithm by anomaly-based attack detection with a feature extraction technique to optimize some attributes for the classification. The experiments were performed on the KDDCUP99 dataset, and results were evaluated by calculating some parameters such as accuracy and speed. After evaluation, the accuracy rate was noted 97.5% for the known attack, and for unknown attack, it was noted as 93.2%.

3.5. Cuckoo Algorithm.

3.5. Cuckoo Algorithm. A cuckoo algorithm is one of the optimal search algorithms inspired by the holoparasite act of cuckoo birds. The birds of these types are not able to complete their reproduction phase by lacking proper host, and these birds can lay their eggs to the nest of the birds that contain eggs that look like them, which means they place their eggs inside the nest of other similar birds. The searching approach followed by the bird is acceptable in different areas for solving different optimization problems. Cuckoo search is applied with three traditional rules: firstly, randomly search location of host nest for placing eggs; secondly, the nest that contains similar eggs as compared to a cuckoo egg; third, the finite number of nests that is considered as 15 for cuckoo search. Hence, the probability P can be taken for its eggs as an object which is represented as $\{P\ (a) \exists a \in (0,\ 1)\}$. The following authors used the cuckoo algorithm to achieve optimization in the attack detection model.

Hao et al. [49] proposed security against denial of service attack by using a cross-layer approach as the best solution. The cross-layer approach was the combined form of device-driver packet filter (cuckoo-based filter) and remotely firewall. Packet filter was designed to filter out abnormal network traffic before it utilizes the resource for higher network protocol layers at a server-side. The performance of the proposed technique was checked through wide-ranging simulated by java and performs better for DDoS attack detection.

3.6. Bacterial Foraging Algorithm. BFO technique is stimulated by a collection of forage behavior of bacteria such as *E. coli* and *M. xanthus*. Particularly, the BFOA algorithm based on the chemotaxis behavior of bacteria can determine chemical gradients and move toward or away from particular signals. The information-conveying process of the algorithm is used to allow cells to collect together swarm to optima. This can be implemented by a sequence of three main processes on a population of replicated cells: first, chemotaxis; second, reproduction; and last, elimination-dispersal. These first steps are responsible for the cost of cells is redefined through the closeness of other cells, and they can move along the modified cost surface at once. The second one in which only those cells are preferred performs best in their whole life that allows being the part of next generation, and in the third one, the cells may discard and low probability new random samples are added.

The following optimization algorithm is efficiently used for cyberattack detection mechanism. Damodaram and Valarmathi [50] applied the bacterial foraging algorithm for the detection of phishing attacks. The traditional systems are intelligent, flexible, and efficient based on association and classification of data mining algorithms, but they are not successful to provide the optimal solution. The proposed model introduced a hybrid optimization algorithm BFOA for achieving an optimal solution for identifying phishing websites. Experimental results were compared with the traditional techniques proved to be very efficient by comparison. Table 1 shows the comprehensive analysis including techniques, datasets, description, and outcomes of different articles from the literature. Table 2 presents the brief description of different metaheuristic techniques with their features and application.

4. Machine Learning Methods

In our day-to-day life, artificial intelligence plays an important role to solve many complex problems. It includes many applications such as speech recognition, language processing, machine intelligence, and fog computing [53, 54]. Machine learning is one of the popular fields of artificial intelligence that is successfully used in solving various computational problems of different areas [56, 57]. Now a days it is extended to more deep networks such as deep learning [58], extreme learning [59], deep extreme learning networks etc.

Machine learning algorithms are classified as “classification,” “clustering,” or “regression.”

This section of the paper discusses various methods of machine learning used in an attack detection system. Here, certain details of these techniques with their results are presented by taking the help of different research papers for each method. In Figure 3, classification of machine learning techniques such as decision trees (DTs), artificial neural networks (ANNs), naive Bayes (NB), and fuzzy set-based approach are referred from the previous literature survey.

The paper presents a detailed study of some important intelligent classification techniques are discussed below.

4.1. Artificial Neural Networks (ANNs). ANN is among the efficiently used systems that stimulated its working like the human brain [1]. ANN works like the human brain which

Table 1: Detailed analysis of comprehensive survey and research articles.

Reference	Technique used	Dataset used	Description	Outcomes
Hao et al. [36]	*The hybrid form of k-means + PSO*	*KDDCUP99*	The proposed model can be used to detect the crowd (undetermined session) is normal or an attack.	The proposed model can detect attacks with better performance.
Momanyi Nyabuga et al. [37]	*Particle swarm optimization*	*KDDCUP99*	The proposed model provides a review and discussions of the denial of service attack detection and prevention mechanisms; moreover, it intended to propose the particle swarm algorithm optimally helps to detect DOS attack.	The simulated outcomes have shown that the proposed PSO-based model was efficiently used for attack detection as compared with other methods.
Shinde and Parvat [26]	*The hybrid form of PSO + SVM*	*NSL-KDD*	The attack detection model was designed using a hybrid form of SVM machine with the PSO technique for the selection of optimal features to achieve high accuracy and performance also lower the FAR alarm than normal IDS.	The hybrid approach of machine learning and optimization technique (ABC-SVM) provides better results than the other single approach. The results showed a detection rate with 98.53% and a false alarm rate with 0.0374.
Siva Sankari1 et al. [39]	*Genetic algorithms*	*KDDCUP99*	The proposed model is designed by using the genetic algorithm (GA) for the detection of DoS.	This detection approach was better-performed attack detection but not proved to be very efficient as comparing its performance with the hybrid technique approached model. However, it provides better results than the traditional one.
Mizukoshi and Munetomo [40]	*Genetic algorithms*	*KDDCUP99*	This proposed model is based on real-time traffic pattern analysis using a genetic algorithm (GA) approach for optimal pattern extraction.	The experimental result has shown that the proposed method performed well as compared with other traditional methods.
Lee et al. [41]	*Genetic algorithms*	*DARPA 2000, LBL-PKT-4*	This proposed model is designed for the detection of distributed denial of service attack using a traffic matrix and optimizes some features of the traffic matrix by using GA.	The detection rate and accuracy by using this method were better compared with other traditional techniques.
Dimitris et al. [43]	*Genetic algorithms*	*KDDCUP99*	This proposed work is designed for the detection of DDoS attacks using a genetic algorithm for efficient feature selection and optimizing some parameters. Genetic algorithm (GA) evaluation used designed error-free neural network detector.	The evaluated results have shown that the features that best qualify for DDoS attack detection were optimally selected by the proposed approach and provide better results.
Chen et.al. [51]	*Ant colony optimization*	*DARPA/ LLDOS KDDCUP99*	This proposed work investigated different complexity of the DDIACS framework and also presents its comparison with the swarm technique and other probability-based techniques.	The results have shown that the proposed framework successfully resolved the problems related to processing attributes, and DDIACS framework provides higher performance than existing methods.
Kumar and Walia [52]	*Ant colony optimization*	*KDDCUP99*	The objective of this work was to design and implement OSLR and DSR protocols for the blackhole attack also prevent the system from the threat.	After evaluation, results showed that the proposed approach performed well on various network performance metrics such as bit error rate, throughput, delay, and packet delivery ratio.
Rais and Mehmood [53]	*Ant colony optimization*	*KDDCUP99*	The proposed model used the ACO optimization technique for better feature selection by various stages of pheromones that help ants to find the optimal features.	Evaluation of the result shows that the proposed approach outperformed in optimal feature selection as compared with the traditional techniques.
Bhuyan et al. [42]	*Artificial bee colony*	*KDDCUP99*	This proposed method is applied to ABC algorithm. Anomaly-based attack detection is used by using different feature selection techniques to minimize the number of unwanted features and pick the best one.	Experimental results have shown that the performance of ABC algorithm was better than traditional approaches and also achieved a high accuracy rate.

Table 2: Comparative study of metaheuristic algorithms.

Features	*PSO*	*GA*	*ACO*	*AIS*	*ABC*	*BFA*
Representation	Dimensional vector for position speed, the best state	Binary, real list of rules, permutation of elements	Undirected graph	Attribute str. (a real-valued vector), integer string, binary string symbolic string	D-dimensional vector (xi = 1, 2, . . ., D)	Represents i-th bacterium at jth chemotactic, k-th reproductive, and l-th dispersal step
Operators	initializer, update, and evaluator	Crossover, mutation, selection, inversion	Pheromone update and measure, trail evaporation	Immune operators cloning, hypermutation and selection based on elitism	Reproduction, replacement of bee, selection	Reproduction, chemotaxis, dispersion, elimination
Datasets for attack detection	*KDDCUP99, DARPA98, NSL-KDD*	*KDDCUP99, DARPA98*	*KDDCUP99, DARPA98*	*KDDCUP99, DARPA98, LLDOS*	*KDDCUP99, DARPA98*	*KDDCUP99, DARPA98*
Permittivity against DoS attack	High	Low	High	Low	Low	Low
Structure and dynamics	Discrete and network components and evolution or learning based	Discrete and network components and evolution based	Discrete components and evolution and learning based	Discrete and network components and evolution or learning based	Discrete and components and evolution based	Discrete and network components and evolution or learning based
Reliability	High	Low	High	High	Low	Low
Time-consuming	Time-consuming because of its complexity but the no. of repetitions is less	Low time-consuming because of its simplicity but the no. of repetitions is high	Time-consuming because of its complexity	Low time-consuming because of its simplicity	Low time-consuming because of its simplicity	Low time-consuming because of its simplicity
Robustness	High	Low as compared to PSO, but more than others	Lower than PSO and GA, more than others	Lower than ACO better	Lowest	Higher than ABS
Parameters	Number of particles, dimension of particles, range of particles, maximum number of iterations, inertia weight	Population size, max generation number, cross-over probability	Number of ants, iterations, pheromone evaporation rate, amount of reinforcement	Population size, no. of antibodies to be selected for hypermutation, number of antibodies to be replaced	No. of food sources which is equal to the no. of employed onlooker bees	The dimension of the search space, number of bacteria, number of steps chemotactic, no. of elimination and dispersal events, no. of reproduction steps, probability
Applications	Power system optimization problems, multimodel problems, multiobjective, dynamic, constrained, and combinatorial optimization problems, anomaly detection, sequential ordering problem, etc.	Pattern recognition, reactive power dispatch, sensor-based robot path planning, multiobjective vehicle routing problem, molecular modeling, web service selection, etc.	Continuous optimization and parallel processing implementations. Vehicle routing problem, graph coloring and set covering, agent-based dynamic scheduling, etc.	Computer security, anomaly detection, clustering/classification, numeric function optimization, virus detection, pattern recognition, etc.	Solving reliability redundancy allocation problem, training neural networks, XOR, decoder-encoder, and 3-bit parity benchmark problems, pattern classification, etc.	Application for harmonic estimation problem in power systems, the parameters of membership functions, and the weights of rules of a fuzzy rule set are estimated, etc.

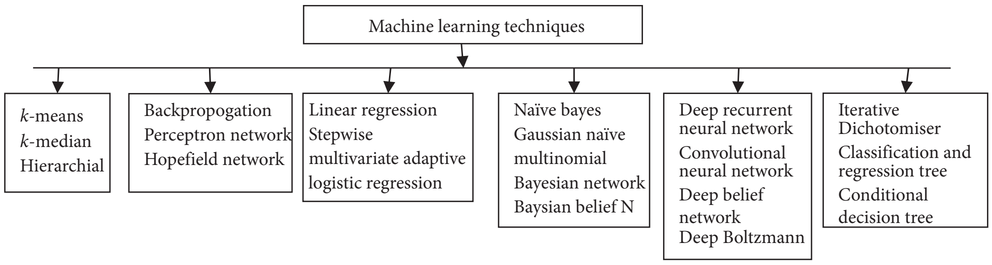


Figure 3: Categories of machine learning techniques.

comprises billions of neurons which are interlinked by different synapses, and its functionality is separated into three major layers which are input, output, and hidden layers in which each connection is associated with some weight. The entire networks are trained and learn from its learning phase and training phase through the weight adjustment, so it enables us to calculate the accurate class to the set of inputs. ANN, as shown in Figure 4, is also defined as a network of numerous computing elements or units that are closely interconnected with each other and also transform a set of inputs to the required outputs. The outcomes are evaluated using the unique weights and elements that are related to each other by interconnectivity between them. The network can generate the desired output by modifying links connecting nodes [60]. Activation function is applied to the set of input nodes, then passed through hidden layer nodes, and finally reaches the output nodes.. ANN works as a well-designed transformation of a set of input to output values. An artificial neural network can work for both the methods of the anomaly- and signature-based attack detection [61].

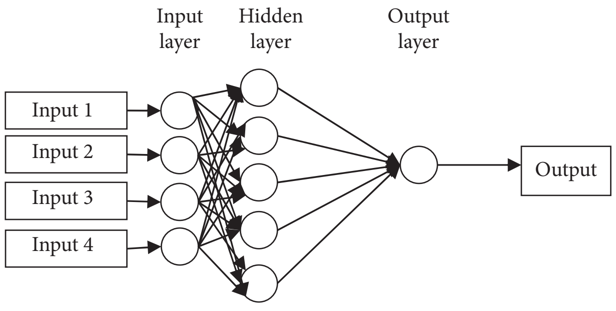


Figure 4: Neural network architecture.

4.1.1. Anomaly-Based Detection Using Artificial Neural Network. Jadidi et al. [28] proposed a model for attack detection using the hybrid form of ANN for detecting attacks by using a flow-based dataset and also applied metaheuristic optimization algorithms for achieving an optimal solution. In this proposed work, there were two hybrid heuristic algorithms such as PSOGSA and cuckoo, which were used to efficiently use the interconnected weights of an MLP network. The resultant network analyzed by flow-based datasets compared its performance with the previously used techniques and found that the proposed hybrid technique enables us to detect attacks with better accuracy.

Jiang et al. [62] proposed a model designed by using hierarchical neural networks for an attack detection system that worked on RBF. The proposed method used the combination of the anomaly- and signature-based detection methods, also having the benefit of the RBF for low training with better accuracy. The RBF anomaly classifier is used for the identification of normal or attack data. Hence, the proposed method enabled us to analyze real-time network traffic.

In Jadidi et al.'s study [63], the proposed model was built on an anomaly-based detection approach which is a very well-known technique and efficiently used for detecting unknown attacks. This work is based on the anomaly-based attack detection method and MLP neural network with a single hidden layer was used. In this attack detection system, GSA was used for the optimization of interconnected weights of a multilayer perceptron network. Hence, the proposed GSA-based detection system successfully achieved 99.43% accuracy.

Ryan et al. [64] proposed a model of intrusion detection designed by using ANN in which the BPNN algorithm is used to model attack detection systems in which the system of some users was used. The dataset used for training and testing was taken from the logs of the UNIX environment. The evaluation of the result found 96% accuracy and a 7% false alarm rate.

4.1.2. The Signature-Based Detection Approach. Cannady [65] proposed a model for intrusion detection that was built by using artificial neural network designed by a multistage classifier approach to detect signature-based (misuse based) detection. The data created by a real-time secure network consist of attack signatures and analyzed approximately thousands of events in which 3000 were simulated attacks.

Nine different features are selected after the data preprocessing step.. Normal or abnormal traffic is recognized by training the system using an artificial neural network, which enables us to learn the collective signatures. The proposed model resulted in 93% accuracy and found efficient after compared with other algorithms.

4.2. Bayesian Network. A Bayesian network is one of the ML techniques that work on the concept of probabilistic graphical model that is represented by some particular variables and the associations between them [66]. The Bayesian network can easily handle incomplete datasets [33]. The network is generally created in the form of a graph where nodes or vertices (V) are used as the random variables and edges (E) as a connecting association between them, and a directed acyclic graph (DAG) is set up. The lower-level nodes are called child nodes that depend on parent nodes or upper-level nodes. Every node or vertices are assigned a random variable and conditional probability [16].

Bayesian classifiers based on Bayes' theorem are used for the classification of the new instances of a data sample named Y. Each instance is a set of attribute values that are denoted as $Y = (y_1, y_2, \ldots, y_n)$. Considering n number of classes, the sample Y is assigned to the class C_i if a given condition is satisfied:

$P(Y \mid C_i)\, P(C_i) > P(Y \mid C_j)\, P(C_j)$ for all i and j in $(1, m)$.

The sample is considered to be the class that has a max probability. In the Bayesian network, the attributes are implicitly conditional independent. Instead of that, naive Bayesian classification provides acceptable outcomes as it focused on the identification of the classes for the instances instead of probabilities. Hence, it can be used in various applications such as text data classification and attack data classification [16].

4.2.1. Anomaly-Based Detection. Panda and Patra [61] had proposed a structure for an attack detection system that used the naïve Bayes algorithm, one of the techniques of machine learning. The experiments applied on a 10% KDDCUP99 dataset, and the system is evaluated by tenfold cross-validations. The experimental results show the proposed approach achieved a higher detection rate than other approaches, the detection rate was noted as 95%, the error rate was 5%, and it was fast and cost-effective.

Farid et al. [67] had proposed representation for intrusion detection where data classification can be done by using one of the popular learning algorithms, naive Bayesian technique. The overall working of the proposed algorithm for intrusion detection had been evaluated on 10% of KDDCUP99. The experimental results founded high accuracy with minimum false positives.

Muda et al. [66] proposed a hybrid method in which the hybridization of two machine learning approaches which are naïve Bayes and k-means clustering technique was used for solving a classification problem. The computational evaluation can be performed on the benchmark KDDCUP99. The proposed model worked with two different phases in the first phase, and the grouping of similar data instances was done according to their behaviors by using the k-means clustering technique. In the second phase, the naïve Bayes classifier was used for classification task and the results are achieved by this approach: the accuracy was noted as 99% and false alarm was less than 0.5%.

Ben Amor et al. [33] proposed a model built on the naïve Bayes classifier and built a normal Bayesian network, and the evaluation is applied on the KDD 1999 dataset and collecting the classes of attacks in the following three major stages for performance measurements. In the first stage, calculate single attack in normal data, in the second stage, contain all four attack types of the KDD 1999 dataset, the problems were resolved by using multiclass classification based on the misuse detection technique, and the third stage consists of normal data and all four attack types using anomaly-based attack detection technique. The experimental results found with better accuracy.

4.2.2. The Signature-Based Detection Approach. Panda and Patra [61] proposed the attack detection model designed by naïve Bayes technique using Weka tool [23], and the experiments were applied on the KDD 1999 dataset that is grouped into different attacks of KDD datasets; finally, the results are compared with the neural network classifier and reported as the naïve Bayes classifier has a high accuracy and false alarm rate than NN.

4.3. Support Vector Machine. SVM is capable of resolving various pattern recognition problems, proposed by Vapnik [25]. SVM uses the concept of supervised learning with related learning algorithms which were mostly applied to signature-based detection in the last few years. It transforms the set of inputs into a high-dimensional space and can be creating an optimal divided hyperplane into the high-dimensional feature space [60]. The SVM classifier is applied to provide improved output for binary classification as compared with further classifiers. SVM promises good performance, and hence, it is used in various fields such as pattern recognition, bioinformatics, text categorization, speaker verification, character recognition, engineering and science, and financial market evaluation [32]. SVM is popular for solving various classification problems because its robustness and efficiently dealing with high-dimensional data also remove the nuisance of the dimensionality problem. SVM was initially designed for binary classification for constructing an optimal hyperplane to maximize the division line among the negative and positive datasets [32].

4.3.1. Anomaly-Based Detection Using Artificial Neural Network. Mukkamala et al. [68] proposed a hybrid model that is the collection of techniques ANN and SVM for the attack detection system. The purpose of using SVM is to achieve better speed and scalability in attack detection system. The experimental results were carried on the DARPA 1998 dataset. The result of the proposed method shown as training time for SVMs is significantly minimum, and it is reported as 17.77 sec shorter than neural networks. The performance of SVM showed that the attack detection system had a higher rate of detection than neural networks.

Chen et al. [69] proposed a model used the combination of set theory and SVM for the attack detection system. The experiments are applied on the KDDCUP99 dataset, and the rough set theory concept is applied at the preprocessing phase and to optimized features. The selection of best features was selected and applied to train the SVM model and accordingly tested. The experimental result has shown that the accuracy was noted as 86.79% and FPR was 29.97%. The accuracy was found better with a reduced false positive rate.

Wang et al. [70] proposed a model for attack detection designed by using a two-hybrid combination of algorithms that worked on improved SVM by using the collective form of PSO and PCA. The experiments were performed on the KDDCUP99 dataset, and principle component analysis (PCA) was used as an effective technique for decreasing dimensions of the dataset. The particle swarm intelligence technique was applied with different parameters in SVM.

The results have shown that the attack detection rate was found to be improved by PCA and PSO combination as compared with PSO-SVM.

Srinoy et al. [21] proposed an attack detection model that is built on a hybrid combination of algorithms which are PSO for SVMs and optimal feature selection, which act as a fitness function of PSO. The evaluation of the result showed that the proposed method successfully recognized both the unknown and known attacks. This proposed technique achieved better classification accuracy by comparing it with different traditional methods.

Shinde and Parvat [26] proposed a model built by the use of PSO and SVM for choosing optimal parameters for gaining a high detection rate and low false alarm rate. Support vector machine (SVM) has the potential for achieving better classification for the attack detection system. The working of SVM is depending on choosing the better parameters. This proposed work used the SVM classifier with the knowledge gain for feature selection. The classified parameters of SVM will be optimally chosen by a PSO algorithm. The experiments were applied on the NSL-KDD dataset, and results show that the proposed method can attain a low false alarm rate and higher detection rate.

Saxena and Richaariya [46] proposed a model which was a hybrid form of the PSO method and SVM. The experiments are applied to the KDDCUP99 dataset. The selection of optimized parameters by binary PSO and classification problem was solved by the support vector machine. The binary PSO provides the finest promising feature subset for creating a better intrusion detection system. The proposed method had to complete that task by following certain major steps which are preprocessing, feature reduction using information gain, and training using hybrid SVM-PSO. Finally, the evaluation has shown that the hybrid combination of PSO and SVM achieved a high detection rate than the simple SVM method.

Wang et al. [71] proposed a model designed by using the SVM-based feature selection algorithm for reducing the dimension of sample data. The anomaly-based intrusion detection technique is used, and MSVM is used with highly optimized parameters by the use of particle swarm optimization (PSO) in the collective form detecting anomalous connections. The experiments were performed on the KDDCUP dataset to measure the efficiency of proposed algorithms (FS-SVM and MSVM-PSO) and the detection precision of MSVM-PSO, also comparing the MSVM-PSO with three different algorithms: these were Bayesian algorithm, *k*-means, and multiclass support vector machine with optimized parameters of the method (MSVM-grid). The experimental results have shown that the hybrid form of MSVM-PSO outperforms than three algorithms in terms of different parameters such as detection rate and accuracy.

4.4. Decision Trees. DT is one of the prominent methods of ML that represented in form of a tree structure where each lower-level node is used to represent a decision or test on the data item which is taken into consideration [1]. The outcome of the test decides the selection of any branch. Classification of any data item is based on a process in which the decision tree algorithms start with its root node and follow the assertions, and the process is carried out until reaching a terminal leaf or node. After reaching the terminal node, a decision is being made. A decision tree is also represented as a unique form of a rule set, categorized by the hierarchical association of rules. A decision tree comprises of following essentials [37]. A decision node which represents a condition or a test on a data item, one of the possible attribute values, or test attribute outcomes is given by branch or edge, and the object belongs to which class is given by the leaf. It starts from the root node of the decision tree and follows the branch indicated by the outcome of each test until a leaf node is reached which is the procedure to classify an object. The leaf node level class is called as unknown object, and the information gaining of the attributes provides the best attribute on the division of subsets.

Ben Amor et al.[28] proposed an attack detection model that was built on the combination of two ML techniques which were DT and Bayesian network. The experiments were applied to the KDDCUP99 dataset and also compared the performance of both the techniques. After the evaluation of the proposed technique, that DT-based method provides much better results than naïve Bayes. If the comparison is based on the computation, then the building of DT is much slower than the Bayesian network. The decision tree selects the optimal features for each node during the creation of the tree based on some defined criteria. The advantage of DT is that it has a good speed of operation and high attack detection accuracy.

Stein et al. [72] proposed the attack detection model which was based on the GA approach for the optimal selection of features used with DT, to achieve maximum detection rate and minimum false alarm rate. The experiments were performed on different attack datasets with different attacks separately. The GA improved some of the categories such as in performance gaining on probe. Hence, it was found that the performance improvement on other attack types is much higher in the testing data.

Karthik et al. [73] proposed the model used three techniques which are used for hybridization, i.e., chi-square, information gain, and relief, and compared their performance using the decision tree classifier. Evaluation can be done by using the KDDCUP99 dataset, and the results have shown that the decision tree which classifies the performance is improved with high accuracy.

4.5. Random Forests. RF is one of the classification techniques that consist of a set of tree-based classifiers [20]. RF is a collection of classification and regression which is invincible in terms of accuracy among the DM techniques. The RF algorithm has numerous applications, used in prediction, probability estimation, and pattern analysis [74]. Random forest classifiers consist of a collection of a huge number of DTs. It is a collection of tree hierarchy in which each tree depends on the values of a random vector that is sampled individually and with the same distribution for all trees in the forest. If new records were taken as input, then random forest makes trees for that records and keeps them in the forest [74].

Malik et al. [75] proposed a hybrid combination of the binary PSO and RF algorithm for the optimal feature selection and classification of attacks in a network. Particle swarm optimization is one of the popular algorithms of the swarm family that has the capability of robust global search and is used for optimal feature extraction, and random forest (RF) is a highly accurate classifier and used for classification. The experiments of the proposed technique are applied to the KDDCUP99 dataset. We also compared the evaluated results of the proposed system with other classifiers, and the final results show that performance achieved by the proposed classifier is much better than the traditional approaches.

Malik and Khan [74] proposed a model designed by using the BMOPSO approach to detect PROBE attacks in the network. The proposed technique focused on two basic parameters to be achieved as first attack DR and second FAR to follow the procedure of feature selection. The experiments were applied to the KDDCUP99 dataset. The proposed approach is used for optimal feature selection from a set of different features and RF techniques used for highly accurate and fast classification. The results have shown that the proposed method performed well for the classification.

4.6. Association Rule and Clustering. Hao et al. [36] proposed an attack detection model for the application level DDoS attack. The weblog record was used for user session extraction and recognizing patterns of the data that are normal or abnormal, and also evaluating the similarities between different sessions. The traditional *k*-means clustering algorithm fails into local optimality. The hybrid collective form of particle swarm optimization *k*-means clustering algorithm (PSO-KMC) was used for constructing an attack detection model. The proposed model enables to detect whether the undetermined sessions are part of DDoS attack or not. The experimental results have shown that the proposed technique can detect attacks effectively with high performance.

Srinoy and Kurutach [76] proposed a model designed by using a data mining-based hybrid approach for attack detection. The algorithm is hybridized by *k*-means and artificial ant clustering algorithm primarily used to generate raw clusters, and they were refined further by *k*-means particle swarm optimization (KPSO). The proposed model had been developed as the evolutionary-based clustering technique. We hybridized the *k*-means algorithm and PSO to find good partitions of the data. The proposed approach allows recognizing not only known attacks but also unknown attacks. After evaluation, the results have shown that the proposed hybrid algorithm performed well with high accuracy.

Ensafi et al. [77] proposed a model designed by using a hybrid combination of algorithms for the attack detection model which were fuzzy logic-based approach and swarm-based approach. The proposed technique is efficiently applied to solve local minima and complex classification problems. The proposed SFK-means approach used the benefits of *k*-means, fuzzy *k*-means, and swarm *k*-means, and all together successfully resolved most of the problems. The importance of the SFK-means algorithm was to overcome local convergence problems in fuzzy *k*-means and the sharp boundary problem in swarm *k*-means. The experiment is applied to the KDDCUP99 dataset and found that the proposed approach was effective in detecting various attacks.

4.7. Hidden Markov Models. Markov chains or hidden Markov models (HMMs) are members of the class of the Markov model. A Markov model consists of set of states that experience some transitions from one state to another by the transition probabilities which decide the logical structure of the model [31]. HMM is one of the machine learning models used in various applications such as speech, pattern, gesture recognition, bioinformatics, and language processing domain. It is also named as a statistical or sequence model in which the system modeling can be done by the Markov process for unknown parameters. The challenging task in the HMM model is to recognize different hidden parameters from the visible parameters [31]. The states of a hidden Markov model represent undetermined conditions that should be modeled and have dissimilar output probability distributions at each state.

4.7.1. Anomaly Detection and Hybrid Detection. Joshi and Phoha [78] proposed the hidden Markov model for intrusion detection. The HMM was used with the following five states and six symbols per state. The states were interconnected in a manner that anyone of state can reach other states. Baum-Welch et al. were applied to evaluate the hidden Markov parameters. The experiments were applied to the KDD 1999 dataset, also evaluates other parameters such as FP and FN rate. The results showed that the accuracy is significantly improved by using more than five features.

4.7.2. Misuse-Based Detection. Ariu et al. [31] proposed an attack detection model for the web applications and used hidden Markov models for the attack signature extraction. The proposed method competitively modeled the classifiers. The experiments were applied to the DARPA 1999 and HTTP dataset. The experiments were applied, and the detection ratewas evaluated higher than 0.8.

4.8. Deep Learning. Deep learning is another field of machine learning that deals with algorithms based on structure and function that resemble the working of the human brain which is called artificial neural networks. It belongs to both the categories of supervised and unsupervised learning and dealing with multilevel representation and features of hierarchical architecture in classification and pattern recognition. Deep learning is popularly used in different applications such as image processing and audio, text, and speech recognition. These techniques are targeted to learn the best feature representation from the bulk amount of unstructured data. Deep learning-based different methods are used to overcome different problems of modeling an efficient attack detection system. There are various deep learning methods which are available such as recurrent neural network (RNN) [79], Boltzmann machine (BM), restricted Boltzmann machine (RBM), deep Boltzmann machine (DBM), deep neural network (DNN) [80], autoencoder, deep/stacked autoencoder, stacked denoising autoencoder, distributed representation, and convolution neural network (CNN). Self-taught learning (STL) [81] is also one of the DL approaches that consist of two stages for the classification. First, the best feature is selected from bulky data, called unsupervised feature learning (UFL). In the second stage, it learned representation of labeled data and used for the classification. DL architectures are similar to a neural network of multiple layers of architecture and various linear or nonlinear functions. The deep learning algorithm that works with high speed and fast learning capability and provides an efficient solution is said to be a successful method.

Niyaz et al. [82] proposed an attack detection model against DDoS attack, a DL-based approach of the multi-vector DDoS detection system in an SDN environment. SDN provides facility to program network devices for accomplished different tasks. The proposed system is designed as a network application over the software-defined network environment controller. The feature selection or classification is done by using deep learning. The result showed high accuracy with a low FP rate for attack detection in the proposed system.

Yin et al. [79] proposed a model designed by using deep learning technique, called the RNN-based intrusion detection system. The performance can be evaluated in the form of binary and multiclass classification, and the number of layers, neurons, and different learning rate was included. Hence, comparison was carried out with different techniques such as ANN, RF, SVM, and other machine learning methods that were previously used by the researchers. The experimental results have shown that the proposed intrusion detection model is suitable for modeling complex classification models that have high accuracy and performance is also superior to the traditional machine learning classification methods in both binary and multiclass classification. The proposed model ensures that improved accuracy of the intrusion detection also provides a better method for intrusion detection.

Javaid et al. [81] proposed a deep learning-based IDS model, using self-taught learning (STL), a DL technique was applied, and the experiment was evaluated on the NSL-KDD dataset for network intrusion. The performance of the proposed approach is better than other traditional approaches of previous work. Comparison can be done on certain parameters which are accuracy, precision, recall, and F-measure values.

Table 3 presents comprehensive study among the machine learning techniques. Table 4 and Table 5 present the performance comparison among machine learning techniques.

5. Datasets Used in Cyberattack Detection

Using machine learning and optimization algorithms for classification and feature selection problems, the dataset is considered to be a very important element. Since these techniques are working with the learning and testing phase in which they learn from the existing data, hence it is essential to have proper knowledge of the dataset that should be used to be aware of how the various authors and scientists may apply different machine learning and optimization algorithms. In this section, we describe different types of datasets used for attack detection in detail by applying machine learning and optimization algorithms. Here, we discuss three broad categories of the dataset which are public datasets, net flow datasets, and packet flow-based datasets.

5.1. Public Datasets. For the attack detection system, there are the following public datasets are discussed in detail. The different public datasets are given in the following sections.

5.1.1. Defense Advanced Research Projects Agency (DARPA 1998). DARPA 1998 is firstly created through the Cyber Systems and Technology Group of MIT Lincoln Laboratory, under Defense Advanced Research Projects Agency and Air Force Research, a laboratory for the assessment of network attack detection systems. DARPA 1998 and 1999 are widely used in various experiments and repeatedly cited in many publications. The DARPA 1998 dataset is formed by the MIT/LL. It creates an interest in the various researchers that may work on different issues of network, base station, and network attack detection system. The evaluation is designed to concentrate on core issues of technology and to motivate much participation to work on security and privacy concerns.

The DARPA 1999 dataset significantly has many attack types as compared to the DARPA 1998 dataset. There were two parts of intrusion detection evaluation of 1999 DARPA: firstly offline evaluation and secondly by a real-time evaluation. The attack detection systems are tested in the offline evaluation mode using network traffic and various audit logs together on network simulators. Some batch mode is applied for processing these data.

5.1.2. Knowledge Discovery Dataset (KDD). The most commonly used benchmark for attack detection is named as the KDD 1999 dataset [66] which was formed for the KDDCUP challenge in 1999. The KDD dataset consists of three major terms which are basic, content, and traffic features and creates 41 attributes (Table 6). The KDD 1999 dataset has some resemblance with the NetFlow dataset, but it is more complicated and has detailed features since the attacks were evaluated.

Another form of NSL-KDD dataset that has 42 attributes (Table 6) was used in this study. NSL-KDD is an enhanced form of the KDD99 dataset on which repeated instances were removed. The NSL-KDD dataset has many different versions in which only 20 percent of training data are used which are determined as KDD train 20% along with 25192 instances. The tested dataset is determined as KDD test that has 22544 instances. Table 6 describes the KDD dataset attributes with class labels. Hence, from these 42 attributes, the 41 can be classified into four different classes mentioned as follows: basic (B) features are of individual TCP connections, content (C) features are inside a connection recommended by domain knowledge, traffic (T) features are processed by two-second

Table 3: Comparative study among different machine learning techniques.

Technique	Principle	Parameters	Advantages	Limitations
k-means	Find out *k* points called centers that are evaluated as the sum of the distances of all points to their respective cluster centers	Cluster center location	High computation, produce closet clusters	Calculation of *K* is a very tough task for a fixed number of clusters. The dissimilarity, the initial and final cluster partition
K-nearest neighbor	The input consists of *k*-closest training of the feature space by using instance-based learning	Class of nearest neighbor	It is easy to implement, less complex	Difficult to deal with arbitrary attributes
Support vector machine	The mapping of input data to the high-dimension space and also dealing with linearly separated data for classification	Features of high dimension	Having high accuracy, flexible and robust in dealing with errors	Takes large time for training, complex to handle learned function (weights)
Hidden Markov model	It is a statistical or sequence-based model that consists set of states, transitions represent the set of possible positions	Pixels in a vision-based input	High-scalable model and easy to understand	Many assumptions about the data. A large number of parameters required to be set. Highly needed training data

Table 4: Performance comparison of different machine learning techniques.

Parameters Accuracy	ANN High	NB High	SVM High	KNN Low	DT Low	RF High	DL High
Training time	High training time due to complexity	Low training time due to simplicity	High training	High training time	High training time due to complex structuring	High training time	High training time complex structuring
Execution time	Average	High	High	Low	Low	Low	High
Large attributes	Dealing well with large attributes	Dealing well with large attributes	Dealing good with large attributes but speed will be very slow	Dealing well with large attributes	Average dealing with larger attributes	Dealing well with large attributes	Dealing good with large attributes but speed will be very slow
Lots of missing attributes	Contradictory	Good performed	Good performed	Low performed	Good performed	Good performed	Good performed
Lots of noisy data	Contradictory	Better dealing with noise	Better dealing with noise	Low capability dealing with noisy data	Average dealing with noise	Average dealing with noise	Better dealing with noise but the overall process is time-consuming
Large datasets	Cannot handle large dataset and speed of processing will be very slow	Better while handling large dataset	Average performed while handling large dataset but processing speed will be very slow	Better while handling large dataset	Average performed	Average performed	Better while handling large dataset but processing speed will be very slow due to complex structure
Detection rate	High	High	High	Low	High	Low	High
Datasets suitable	KDD99 and NSL-KDD	KDD99 and NSL-KDD	NSL-KDD, KDDCUP99, and DARPA	NSL-KDD, KDDCUP99, and DARPA	KDDCUP99	KDDCUP99	KDD99 and NSL-KDD

time window, and host (H) features are planned to assess attacks that last for more than two seconds.

5.2. Packet Flow-Based Dataset. Over the Internet, there are the following broadly used protocols, for example, IP, ICMP, IGMP, TCP, and UDP. Hence, due to the client programs, running these protocols can create huge traffic over the network. The overall incoming and outgoing packets use the physical interfaces such as Ethernet port for transmission and reception of the packets. Because at the network layer, none of the abovementioned protocols is directly transferred to its lower layer, hence these protocols are encapsulated inside the data field of the IP packet format, and then it can be transferable to its lower layer. In the data link layer or MAC

Table 5: Complexities of various machine learning techniques.

Algorithm	Complexity	Capability	References
ANN	O (emnk)	Low	e = no. of epoch, k = no. of neuron
Naive Bayesian	O (mn)	High	m = no. of training, n = no. of feature
SVM	R^3 & nS	High	R = no. of free support vector
Decision tree	O (mn^2)	Medium	M = no. of features
Clustering k-means	O (kmni)	High	I = no. of iteration, k = no. of cluster
Clustering hierarchical	O (n^3)	High	N = no. of data points
Association rules	O (n^2)	Low	N = no. of input transactions

Table 6: Detail of KDD dataset attributes with different classes.

S. no.	Dataset feature name	Classes
1	Duration	Basic
2	protocol_type	Basic
3	Service	Basic
4	src_bytes	Basic
5	dst_bytes	Basic
6	Flag	Basic
7	Land	Basic
8	wrong_fragme nt	Basic
9	Urgent	Basic
10	Hot	Content
11	num_failed_lo gins	Content
12	logged_in	Content
13	num_compro missed	Content
14	root_shell	Content
15	su_attempted	Content
16	num_root	Content
17	num_file_crea tions	Content
18	num_shells	Content
19	num_access_fi less	Content
20	num_outboun d_cmds	Content
21	is_hot_login	Content
22	is_guest_login	Content
23	Count	Traffic
24	serror_rate	Traffic
25	rerror_rate	Traffic
26	same_srv_ rate	Traffic
27	diff_srv_r ate	Traffic
28	srv_count	Traffic
29	srv_serror_rate	Traffic
30	srv_rerror_rate	Traffic
31	srv_diff_h ost_rate	Traffic
32	dst_host_count	Host
33	dst_host_srv_co unt	Host
34	dst_host_same_ srv_rate	Host
35	dst_host_diff_sr v_rate	Host
36	dst_host_same_src_port_rate	Host
37	dst_host_count	Host
38	dst_host_srv_co unt	Host
39	dst_host_same_ srv_rate	Host
40	dst_host_diff_sr v_rate	Host
41	dst_host_srv_re rror_rate	Host
42	Class	Host

layer, an Ethernet frame consists of about 1500 bytes of the payload, and this consists of encapsulated IP payload with IP header and IP packet contains itself its header and in its data section higher-level protocols such as HTTP, NFS, POP, telnet, and TFTP.

5.3. NetFlow Data. It contains the detail of the router and its features, and the routers have the capability to collect IP packet traffic as these packets are in and out from the router. The NetFlow generally Cisco's version 5 can introduce the network flowing as the sequential flow of packets in the same direction and defines seven types of attributes which are as follows: source and destination IP address, IP protocol, source and destination port, interface, and IP type of service.

6. Observations and Evaluations

The overall performance of any attack detection system is done by evaluating the performance of the various techniques that are applied to it and by the help of some parameters. In this literature survey, it is observed that the KDDCUP and DARPA benchmarks are very suitable and highly used for the evaluation of the performance of the system by applying different metaheuristic and machine learning techniques. In this study, different machine learning and metaheuristic techniques were not applied to build any IDS system but applied to certain cyber data for evaluating their performance. The major parameters that were calculated by applying different techniques on the cyber data are accuracy, detection rate, false alarms, detection rate, false positive, false negative, etc. These parameters show that the ability of a particular technique is suitable for attack detection or not. By the study of different research papers, a certain comparison is shown based on parameters and different techniques of metaheuristic and machine learning on the various cyber data. Figures 5–7 show a comparison among various ML and MH techniques applied on different datasets by different researchers and detection rate, accuracy, and FAR parameter evaluated.

In Figures 5–8, the comparison on the basis of different criteria among previous research studies that used ML and MH techniques in cyberattack detection is shown. It is found that ML and MH techniques improve performance in cyberattack detection models. In Figure 5, we have used previous papers and showed improved detection rate of the different models by ML and MH techniques. In Figure 6, we highlight another important parameter accuracy that is also seen to be improved by using ML and MH techniques. In Figure 7, we highlight the performance of algorithm on the FAR parameter that is also seen to be improved, and finally, in Figure 8, we have shown the usability of benchmarks that are popular in cyberattack detection. Hence, in this research, we present the successful usability of ML and MH

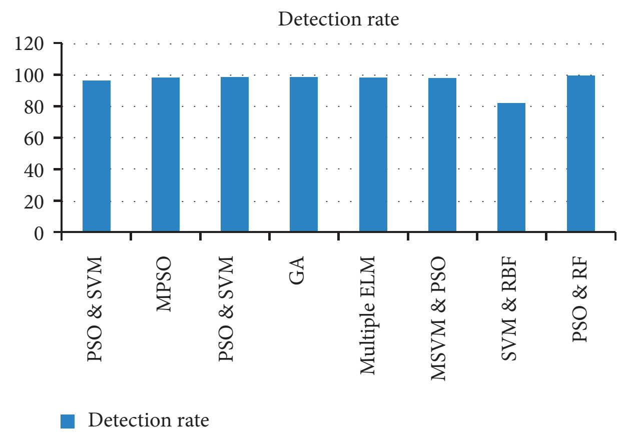


FIGURE 5: Comparison among different ML and MH techniques on the basis of detection rate from Table 7.

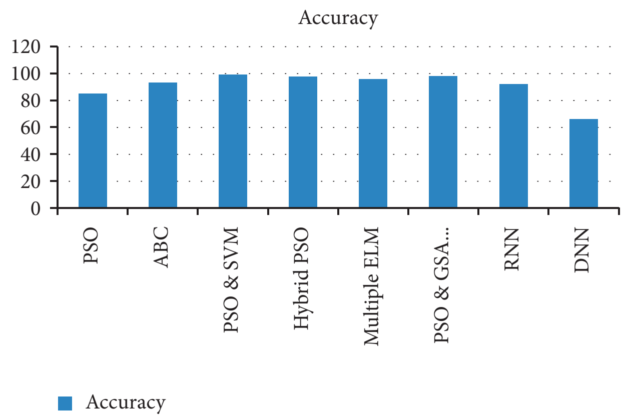


FIGURE 6: Comparison among different ML and MH techniques on the basis of accuracy from Table 7.

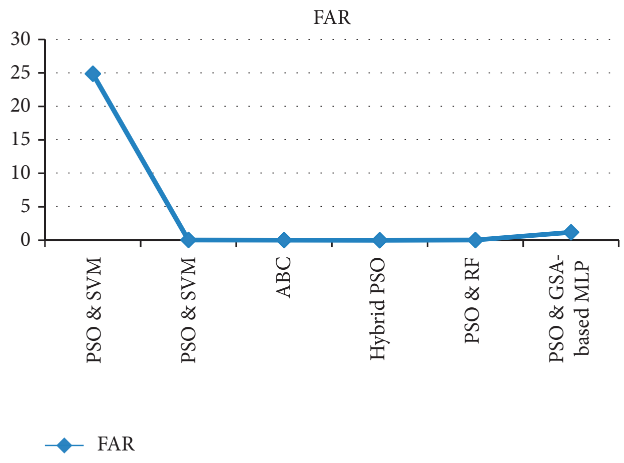


FIGURE 7: Comparison among different ML and MH techniques on the basis of FAR from Table 7.

techniques in cybersecurity domain, and furthermore, these techniques will be explored to perform well in this domain.

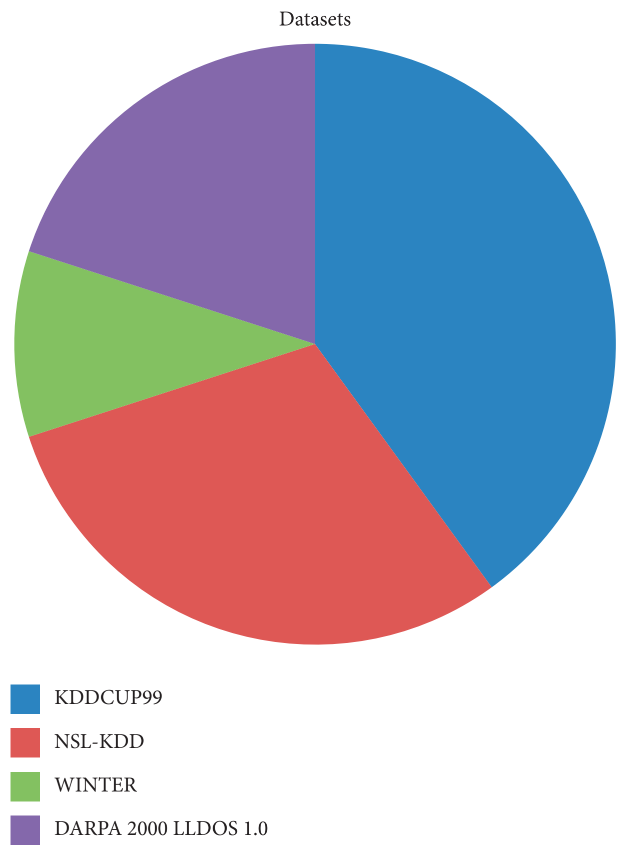


FIGURE 8: Comparison among different ML and MH techniques of previous research studies based on datasets from Table 7.

Table 7 presents the comparative study on the basis of performance among various literature articles and techniques.

7. Challenging Issues and Future Directions

Here, we discuss the different challenges of machine leaning as well as optimization algorithms as follows.

There are two types of problem are categorized while talking about machine learning: one is the regression and the other is classification. The basic difference between both the approaches is about its output value (either continuous or discrete). In cybersecurity, mostly problems are associated with classification which provides categorical output. In order to design the model that can classify unknown data, it is important to first train the network using representative examples. . This phase is usually called either as training. To achieve the same, robust technique of learning will be take into consideration. For examples, the commonly used techniques are SVM [20], decision tree [72, 73], naïve Bayes [20], etc. The input data are an important factor for the learning techniques because they need preprocessing to meet the specification of techniques [87]. The important solutions for betterment in leaning algorithms in performance prospective are as follows:

(i) Dataset that is used for training should be labeled in case of output-based learning

Table 7: Comparative study of the performance of different research citations.

References	Detection rate (%)	Accuracy (%)	Far	Technique	Datasets
[20]		98.70	95.60%	SVM, naïve Bayesian	MIT Lincoln Lab, IDS
[21]	96.11	—	24.88	PSO, SVM	KDDCUP99
[23]	—	—	—	AIS	nit DARPA LLDOS 1.0
[24]	98	—	—	MPSO	KDDCUP
[26]	98.53	—	0.0374	PSO, SVM	NSL-KDD
[27]	95	95	—	BARTD, cuckoo	KDDCUP
[28]	—	98.2/99.55	1.19/ 0.21	MLP, PSO, GSA, cuckoo	DARPA and WINTER
[72]	98.4/96	—	—	GA	DARPA and WINTER
[25]	96.7	—	—	PSO	KDDCUP
[69]	—	85.13	—	PSO	LLDOS
[83]	—	93.25	0.02	ABC	KDDCUP
[46]	—	99.4	—	PSO, SVM	KDDCUP
[32]	—	99.25	0.75	PSO, SVM	PMU2015
[55]	—	84.29	—	Multiple ELM	NSL-KDD
[71]	97.64	—	—	MSVM, PSO	KDDCUP
[21]	—	97.7	0.002	Hybrid PSO	KDDCUP
[77]	96.11	—	3.89	PSO-SVM	KDDCUP99
[74]	99.92	—	0.029	PSO RF	KDDCUP99
[84]	—	89.6	—	RNN	NSL-KDD
[84]		92		LSTM	NSL-KDD
[85]		93.20, 78.1, 66, 96.6		DNN	KDDCUP99, NSL-KDD, UNSW-NB15, WSN-DS
[86]	81.8	—	—	SVM RBF	KDDCUP

(ii) The sample instances while training must represent all classes of the model

(iii) Identify the algorithm with better learning function, and train the model and regulate the parameters using separate data

(iv) Evaluate the model on dissimilar data such as test data

Now, selection of features is a most significant step for achieving better input representation. There are numerous methods, such as filter based, correlation based, wrapping, and heuristic, which were used. Here, we have discussed about metaheuristic techniques that are popularly used in cybersecurity basically for attack datasets. Metaheuristic methods provide two types of solutions such as single-based or population-based solutions. The population-based solution is mostly used in cyberattack problems due to providing multiple solutions. The most commonly used algorithm are already discussed in this paper such as PSO [21, 26, 38, 88], GA [68], ACO [21, 45], ABC [46], etc.

Generally, in cybersecurity, the attack detection problem requires high-level solution methods (metaheuristic methods) that enable us to escape from local optima and execute a robust search of a solution space. However, these methods are unable to perform with large-size or multidimensional datasets and sometimes suffer with convergence problems. Cybersecurity deals with the high-dimensional data as attack datasets are too large to handle. Hence, the advanced technique of learning comes into picture to deal with high-dimensional data. The advanced learning techniques such as deep learning methods [79, 81, 82] and latest artificial intelligence techniques can provide better solutions for cyberattack detection problem by efficiently covering the classification as well as feature selection problems [89–92].

8. Conclusion and Discussion

Cyberattack is one of the challenging areas of research. This study provides imminent research studies in the field of cyberattack detection, a summary of the different techniques related and work done in recent years.

In this paper, more than eighty recent related and fine publications of different conferences and journals were used, which highlights the previous study of different metaheuristic algorithms (MHs) and machine learning (ML) techniques in the attack detection system. ML and MH performance presents in comparative study in Table 1 and Table 2. In Table 1, we found out that these techniques provide better outcomes; hence there is need of exploring application of such techniques in other fields of cybersecurity.. In Table 2, we discuss about the internal properties of algorithms for their better use in computation.

Initially, the most important step is finding example papers that provide a detailed explanation of different machine learning (ML) and metaheuristic (MH) methods in the cybersecurity environment, for both the signature and anomaly-based detection. So, the analysis of the detailed survey presented in the paper states the fact that the machine learning and optimization techniques are more preferred, but regrettably, techniques that provide maximum efficiency and performances had not been invented yet, it is very difficult to provide one recommendation for each technique, and depending on the type of attack, the system is made-up to detect. For evaluating the performance and effectiveness of any techniques, there are several criteria which are available, and it cannot be decided by taking some of them into account. The parameters that were evaluated are listed as accuracy, detection rate, the time complexity for classifying an unknown instance with a trained model, and understanding of the final solution of each machine learning and metaheuristic technique. One more critical characteristic of machine learning and metaheuristic algorithms is a type of dataset for the training and testing of systems in the attack detection process that should be carefully selected and preceded. Hence, the potential use of ML and MH techniques for the computation of the attack detection system is inspiring the advances required to realize the reliable, efficient, accurate, and robust attack detection systems.

Abbreviations

ML:	Machine learning
MH:	Metaheuristic
DoS:	Denial of service attack
DDoS:	Distributed denial of service attack
GA:	Genetic algorithm
ACO:	Ant colony optimization
PSO:	Particle swarm optimization
AIS:	Artificial immune system
ABC:	Artificial bee colony
BFA:	Bacterial foraging algorithm
ANN:	Artificial neural network
NB:	Naïve Bayes
KNN:	K-nearest neighbor algorithm
SVM:	Support vector machine
DT:	Decision tree
RF:	Random forest
DL:	Deep learning
HMM:	Hidden Markov model
KDD:	Knowledge discovery database.

Data Availability

No data were used to support the findings of this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. Ganapathy, K. kulothungan, S. Muthurajkumar, M. Vijayalaxami, P. Yogesh, and A. kannan, "Intelligent feature selection and classification techniques for intrusion detection in networks: a survey," *Journal of Wireless Networking and Communications*, vol. 271, 2013.

[2] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, 2016.

[3] O. Adil Mahdi, Y. R. Bahar Al-Mayouf, A. Basil Ghazi, A. W. Abdul Wahab, and M. Y. I. B. I. Idna Bin Idris, "An energy-aware and load-balancing routing scheme for wireless sensor networks," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 12, no. 3, pp. 1312–1319, 2018.

[4] S. Deol and L. Kaur, "Review on detection and prevention schemes for flooding attack in WSNS," *International Journal of Advance Research Ideas and Innovations in Technology*, vol. 3, 2017.

[5] C. Borrego, M. Amadeo, A. Molinaro, and R. H. Jhaveri, "Privacy-preserving forwarding using Homomorphic encryption for information-centric wireless Ad Hoc networks," *IEEE Communications Letters*, vol. 23, no. 10, pp. 1708–1711, 2019.

[6] S. Venkatraman and M. Alazab, "Use of data visualisation for zero-day Malware detection," *Security and Communication Networks*, vol. 2018, Article ID 1728303, 13 pages, 2018.

[7] R. U. Khan, X. Zhang, R. Kumar et al., "An adaptive multilayer botnet detection technique using machine learning classifiers," *Applied Sciences*, vol. 9, no. 11, p. 2375, 2019.

[8] A. Karim, S. Azam, B. Shanmugam, K. Kannoorpatti, and M. Alazab, "A comprehensive survey for intelligent spam email detection," *IEEE Access*, vol. 7, pp. 168–295, 2019.

[9] R. M., S. P. Maddikunta, P. K. R. Parimala et al., "An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture," *Computer Communications*, vol. 160, pp. 139–149, 2020.

[10] B. A. Khalaf, S. A. Mostafa, A. Mustapha, M. A. Mohammed, and W. M. Abduallah, "Comprehensive review of artificial intelligence and statistical approaches in distributed denial of service attack and defense methods," *IEEE Access*, vol. 7, pp. 51691–51713, 2019.

[11] P. Dixit and S. Silakari, "Deep learning algorithms for cybersecurity applications: a technological and status review," *Computer Science Review*, vol. 39, p. 100317, 2021.

[12] A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, and K. Kifayat, "A comprehensive survey of AI-enabled phishing attacks detection techniques," *Telecommunication Systems*, vol. 76, no. 1, pp. 139–154, 2020.

[13] M. A. Mohammed, S. S. Gunasekaran, S. A Mostafa, A. Mustafa, and M. K. Abd Ghani, "Implementing an agent-based multi-natural language anti-spam mode," in *Proceedings of the International Symposium on Agents, Multi-Agent Systems and Robotics (ISAMSR)*, pp. 1–5, Putrajaya, Malaysia, August 2016.

[14] C. J. Tu, L. Y. Chuang, J. Y. Chang, and C.-H. Yang, "Feature selection using PSO-SVM," *IAENG International Journal Of Computer Science (IJCS)*, vol. 33, pp. 138–143, 2006.

[15] M. Athari and K. Borna, "Using meta heuristic algorithms of genetic, particle swarm optimization and glowworm in the intrusion detection system," *International Journal of Computer Science and Network Security*, vol. 16, no. 10, 2016.

[16] R. D. Sagarin and T. Taylor, "Natural Security: how biological systems use the information to adapt in an unpredictable world," *Information security*, vol. 14, 2012.

[17] S. Jamali and G. Shaker, "PSO-SFDD: defense against SYN flooding DoS attacks by employing PSO algorithm," *Computers & Mathematics with Applications*, vol. 63, no. 1, pp. 214–221, 2012.

[18] M. Tarao and T. Okamoto, "Toward an artificial immune server against cyber attacks: enhancement of protection against DoS attacks," *Procedia Computer Science*, vol. 96, pp. 1137–1146, 2016.

[19] S. Bhattacharya, S. R. K. S, P. K. R. Maddikunta et al., "A Novel PCA-Firefly based XGBoost classification model for Intrusion Detection in Networks using GPU," *Electronics*, vol. 9, no. 2, p. 219, 2020.

[20] J. Visumathi and K. L. Shunmuganathan, "Improved detection of dos attacks using intelligent computation techniques," *SRIMCA*, vol. 3, no. 2, 2010.

[21] S. Srinoy, "Intrusion detection model based on particle swarm optimization and support vector machine," in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications*, Honolulu, HI, USA, April 2007.

[22] S. Mourougan and M. Aramudhan, "Hybrid evolutionary algorithm based intrusion detection system for denial of service attacks," *Indian Journal of Science and Technology*, vol. 8, no. 35, 2016.

[23] U. Akyazı, A. Şima Uyar, *Detection of DDoS Attacks via an Artificial Immune System-Inspired Multiobjective Evolutionary Algorithm*, pp. 1–10, Springer, Berlin, Germany, 2010.

[24] B. Ben Sujitha and V. Kavitha, "Layered approach for intrusion detection using multiobjective particle swarm optimization," *International Journal of Applied Engineering Research*, vol. 10, no. 12, pp. 31999–32014, 2015.

[25] H. Zheng and M. Hou, ""Yu Wang," an efficient hybrid clustering-PSO algorithm for anomaly intrusion detection," *Journal of Software*, vol. 6, no. 12, 2011.

[26] P. Shinde and T. Parvat, "Analysis on intrusions detection based on support vector machine optimized with swarm intelligence," *International Journal of Computer Science and Mobile Computing IJCSMC*, vol. 3, pp. 559–566, 2015.

[27] K. Munivara Prasad, A. Rama Mohan Reddy, and K. Venugopal Rao, "BARTD: Bio-inspired anomaly-based real-time detection of under rated App-DDoS attack on web," *Journal of King Saud University Computer and Information Sciences*, vol. 32, 2017.

[28] Z. Jadidi, V. Muthukkumarasamy, and E. Sithirasenan, "Metaheuristic algorithms based flow AnomalyDetector," in *Proceedings of the APCC*, Bali, Indonesia, August 2013.

[29] K. P. Mohan Kumar and M. Aramuthan, "Hybrid Network Intrusion Detection for DoS Attacks," *IJCTA*, vol. 9, pp. 15–22, 2016.

[30] Y. Wang and C. Wang, "Based on the ant colony algorithm is a distributed intrusion detection method," *International Journal of Security and Its Applications*, vol. 9, no. 4, pp. 141–152, 2015.

[31] D. Ariu, R. Tronci, and G. Giacinto, "HMMPayl: an intrusion detection system based on Hidden Markov Models," *Computers & Security*, vol. 30, no. 4, pp. 221–241, 2011.

[32] C. Kolias, G. Kambourakis, and M. Maragoudakis, "Swarm intelligence in intrusion detection: a survey," *Computers & Security*, vol. 30, 2011.

[33] N. Ben Amor, S. Benferhat, and Z. Elouedi, "Naive bayesian networks in intrusion detection systems," in *Proceedings of the ACM Symposium on Applied Computing (SAC)*, Nicosia, Cyprus, March 2004.

[34] S. M. H. Bamakan, B. Amiri, and M. M. Yong Shi, "A new intrusion detection approach using PSO based multiple criteria linear programming," *Information Technology and Quantitative Management*, vol. 55, pp. 231–237, 2015.

[35] S. Khajouei Nejad, S. Jabbehdari, and M. H. Moattar, "A hybrid intrusion detection system using particle swarm optimization for feature selection," *International Journal of Soft Computing and Artificial Intelligence*, vol. 3, no. 2, 2015.

[36] X. Hao, B. Meng, and K. Gu, "Detecting DDoS attack based on PSO Clustering algorithm," in *Proceedings of the 3rd International Conference on Materials Engineering, Manufacturing Technology and Control (ICMEMTC 2016)*, Taiyuan, China, February 2016.

[37] S. Momanyi Nyabuga, W. Cheruiyot, and M. Kimwele, "Using particle swarm optimization (PSO) algorithm to protect vehicular Ad Hoc networks (VANETS) from denial of service (DOS) attack," *IJARCET*, vol. 5, 2016.

[38] W. Guoli, "Traffic prediction and approach based on PSO optimized Elman neural network," in *Proceedings of the 2019 11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, Qiqihar, China, April 2019.

[39] L. K. Siva Sankari, D. C. Joy Winnie Wise, and B. Priya, "An efficient method for denial of service attack detection using genetic algorithm," *IJARSE*, vol. 4, 2015.

[40] M. Mizukoshi and M. Munetomo, "Distributed denial of services attack protection system with genetic algorithms on hadoop cluster computing framework," in *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC) IEEE*, Sendai, Japan, May 2015.

[41] J. H. Lee, D. S. Kim, S. M. Lee, and J. S. Park, "Ddos attacks detection using GA based optimized traffic matrix," in *Proceedings of the Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, Washington, DC, USA, June 2015.

[42] M. H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya, and J. K. Kalita, "Detecting distributed denial of service attacks: methods, tools and future directions," *The Computer Journal*, vol. 57, no. 4, pp. 537–556, 2014.

[43] G. Dimitris, T. Ioannis, and D. Evangelos, "Feature selection for robust detection of distributed denial-of-Service attacks using genetic algorithms," in *SETN, Lecture Notes in Computer Science*, vol. 3025, Springer, Berlin, Germany, 2004.

[44] S. M. Lee, D. S. Kim, J. H. Lee, and J. S. Park, "Detection of DDoS attacks using optimized traffic matrix," *Computers & Mathematics with Applications*, vol. 63, no. 2, pp. 501–510, 2012.

[45] M. Aldwairi, Y. Khamayseh, and Mh. A. Masri, "Application of artificial bee colony for intrusion detection systems," *Security and Communication Networks*, vol. 8, 2012.

[46] H. Saxena and V. Richaariya, "Intrusion detection in KDD99 dataset using SVM-PSO and feature reduction with information gain," *International Journal of Computer Applications*, vol. 98, no. 6, 2014.

[47] V. V. Mahale and D. B. Gothawal, "Cuckoo filter & Remote firewall: a mechanism for mitigation of distributed denial of service attacks," *International Research Journal of Engineering and Technology (IRJET)*, vol. 3, no. 6, 2016.

[48] V. Priyadharshini and K. Kuppusamy, "Prevention of DDOS attacks using new cracking algorithm," *Engineering Research and Applications (IJERA)*, vol. 2, no. 3, pp. 2263–2267, 2012.

[49] X. Hao, B. Meng, and K. Gu, "Detecting DDoS attack based on PSO Clustering algorithm," in *Proceedings of the ICMEMTC*, Taiyuan, China, February 2016.

[50] R. Damodaram and M. L. Valarmathi, "Bacterial foraging optimization for fake website detection," *International Journal of Computer Science & Applications (TIJCSA)*, vol. 1, no. 11, 2013.

[51] H. H. Chen and S. K. Huang, "LDDoS attack detection by using ant colony optimization algorithms," *Journal of Information Science & Engineering*, vol. 32, no. 4, 2016.

[52] N. Kumar and L. Walia, "Ant colony optimization based approach for the detection of black Hole attack in WANET," *IJCSMC*, vol. 4, no. 6, pp. 469–480, 2015.

[53] H. M. Rais and T. Mehmood, "Dynamic ant colony system with three-level update feature selection for intrusion

detection," *International Journal of Network Security*, vol. 20, no. 1, pp. 184–192, 2018.

[54] K. H. Abdulkareem, M. A. Mohammed, S. S. Gunasekaran et al., "A review of Fog computing and machine learning: concepts, applications, challenges, and open issues," *IEEE Access*, vol. 7, pp. 153123–153140, 2019.

[55] M. Latah and L. Toker, *An Efficient Flow-Based Multi-Level Hybrid Intrusion Detection System for Software-Defined Networks*, Springer, Berlin, Germany, 2018.

[56] K. Chandra, G. Kapoor, R. Kohli, and A. Gupta, "Improving software quality using machine learning," in *Proceedings of the 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*, pp. 115–118, Greater Noida, India, February 2016.

[57] A. R Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. U. haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," in *Proceedings of the IEEE Transactions on Intelligent Transportation Systems (Early Access)*, IEEE, New York , NY, USA, December 2020.

[58] M. Mittal, C. Iwendi, S. Khan, and A. R Javed, "Analysis of security and energy efficiency for shortest route discovery in low-energy adaptive clustering hierarchy protocol using Levenberg-Marquardt neural network and gated recurrent unit for intrusion detection system," *Transactions on Emerging Telecommunications Technologies*, 2020.

[59] H. Guang-Bin, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[60] D. Chavan, C. Francis, E. M. Thomas, and P. Moraye, "Comparative study of preventive algorithms of Ddos attack," *International Journal of Scientific & Engineering Research*, vol. 7, no. 2, 2016.

[61] M. Panda and M. R. Patra, "network intrusion detection using naïve bayes," *International Journal of Computer Science and Network Security*, vol. 7, no. 12, 2007.

[62] J. Jiang, C. Zhang, and M. Kame, "RBF-based real-time hierarchical intrusion detection systems," *International Joint Conference on Neural Networks*, vol. 2, pp. 1512–1516, 2003.

[63] Z. Jadidi, V. Muthukkumarasamy, and M. Sheikhan, "Flow-based anomaly detection using neural network optimized with GSA algorithm," in *Proceedings of the International Conference on Distributed Computing Systems Workshops*, Mesa, Arizona, April 2013.

[64] J. Ryan, M.-J. Lin, and R. Miikkulainen, "Intrusion detection with neural networks," *Advances in Neural Information Processing Systems*, pp. 943–949, 1998.

[65] J. Cannady, "Artificial neural networks for misuse detection," in *Proceedings of the National Information Systems Security Conference (NISSC'98)*, pp. 443–456, Arlington: Virginia Press, October 1998.

[66] Z. Muda, W. Yassin, M. N. Sulaiman, and N. I. Udzir, "K-means clustering and naive bayes classification for intrusion detection," *Journal of Information Technology and Applications*, vol. 4, no. 1, pp. no. 13–25, 2014.

[67] D. M. Farid, M. Zahidur Rahman, and C. Mofizur Rahman, "Adaptive intrusion detection based on boosting and naïve bysian classifier," *International Journal of Computer Applications*, vol. 24, no. 3, 2011.

[68] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection: support vector machines and neural networks," *IJCNN*, vol. 2, 2002.

[69] R.-C. Chen, K.-F. Cheng, Y.-H. Chen, and C.-F. Hsieh, "Using Rough set and support vector machine for network intrusion detection system," in *Proceedings of the Asian Conference on Intelligent Information and Database Systems*, Quang binh, Vietnam, April 2009.

[70] H. wang, G. zhang, E. Mingjie, and N. Sun, "A novel Intrusion detection method based on improved SVM by Combaining PCA and PSO," *Wuhan University Journal of Natural Science*, vol. 16, 2011.

[71] G. Wang, S. Y. Chen, and J. Liu, "Anomaly-based intrusion detection using multiclass-SVM with parameters optimized by PSO," *International Journal of Security and Its Applications*, vol. 9, no. 6, pp. 227–242, 2015.

[72] G. stein, B. chen, and A. S. Wu, ""Kien A hua" Decision tree classifier for network intrusion detection with GA based feature selection," in *Proceedings of the 43rd Annual Southeast Regional Conference*, Melbourne, FL, USA, March 2005.

[73] R. Karthik, S. Veni, and B. L. Shivakumar, "Network intrusion detection using feature selection and decision tree classifier," in *Proceedings of the TENCON*, Osaka, Japan, November 2008.

[74] A. J. Malik and F. A. Khan, "A hybrid technique using multi-objective particle swarm optimization and random forests for PROBE attacks detection in a network," in *Proceedings of the International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2473–2478, IEEE, Bari, Italy, October 2013.

[75] A. J. Malik, W. Shahzad, and F. A. Khan, "Binary PSO and Random Forests Algorithm for PROBE Attacks Detection in a Network," in *Proceedings of the 2011 IEEE Congress of Evolutionary Computation (CEC)*, New Orleans, LA, USA, June 2011.

[76] S. Srinoy and W. Kurutach, "Combination artificial ant clustering and K-PSO clustering approach to network security model," in *Proceedings of the International Conference on Hybrid Information Technology (ICHIT'06)*, Cheju Island, Korea, November 2006.

[77] R. Ensafi, S. Dehghanzadeh, and M. R. Akbarzadeh, "Optimizing Fuzzy K-means for network anomaly detection using PSO," in *2008 IEEE/ACS International Conference on Computer Systems and Applications*, Doha, Qatar, April 2008.

[78] S. S. Joshi and V. V. Phoha, "Investigating hidden Markov models capabilities in anomaly detection," in *Proceedings of the 43rd Annual Association For Computing Machinery Southeast Conference, ACMSE*, New York, NY, USA, June 2005.

[79] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, 2017.

[80] A. Rehman, S. U. Rehman, M. Khan, M. Alazab, and T. Reddy G, "CANintelliIDS: Detecting In-vehicle intrusion attacks on a controller area network using CNN and attention-based GRU," in *Proceedings of the IEEE Transactions on Intelligent Transportation Systems (Early Access)*, IEEE, New York , NY, USA, December 2021.

[81] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning Approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-Inspired Information and Communications Technologies*, New York, NY, USA, December 2016.

[82] Q. Niyaz, W. Sun, and Y Ahmad, "A deep learning based DDoS detection system in Software Defined Networking (SDN)," *EAI Endorsed Transactions on Security and Safety*, vol. 4, 2017.

[83] Q. Qian, J. Cai, and R. Zhang, "Intrusion detection based on neural networks and artificial bee colony algorithm," in *Proceedings of the 2014 IEEE/ACIS 13th International*

Conference on Computer and Information Science (ICIS), IEEE, Taiyuan, China, June 2014.

[84] L. Thi-Thu-Huong, Y. Kim, and H. Kim, "Network intrusion detection based on novel feature selection model and various recurrent neural networks," *Applied Science*, vol. 9, p. 1392, 2020.

[85] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, 2019.

[86] U. Ravale, N. Marathe, and P. Padiya, "Feature selection based hybrid anomaly intrusion detection system using K means and RBF Kernel function," *Procedia Computer Science*, vol. 45, 2015.

[87] A. Q. Lima and B. Keegan, "Challenges of using machine learning algorithms for cybersecurity: a study of threat classification models applied to social media communication data, Cyber Influence and Cognitive Threats," *Cyber influence and Cognitive Threats*, vol. 2020, pp. 33–52, 2019.

[88] B. Tan, Y. Tan, and Y. Li, "Research on intrusion detection system based on improved PSO-SVM algorithm," *Chemical Engineering Transactions*, vol. 51, 2016.

[89] Y. F. Hernandez-Julio, I. Merino-Fuentes, R. R. Gonzalez-Diaz, A. Guerrero-Avendano, L. V. O. Toledo, and W. N. Bernal, "Fuzzy knowledge discovery and decision-making through clustering and Dynamic tables: application in Colombian business Finance," in *Proceedings of the 2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, Seville, Spain, June 2020.

[90] J. M. Saiz-Álvarez, A. Vega-Muñoz, Á. Acevedo-Duque, and D. Castillo, "B corps: a socioeconomic approach for the COVID-19 post-crisis," *Frontiers in Psychology*, vol. 11, no. 1867, 2020.

[91] R. H. Jhaveri, A. Desai, A. Patel, and Y. Zhong, "A sequence number prediction based bait detection scheme to mitigate sequence number attacks in MANETs," *Security and Communication Networks*, vol. 2018, Article ID 3210207, 13 pages, 2018.

[92] S. Ramani, R. Jhaveri, and C. Borrego, "Applications in security and evasions in machine learning: A survey," *Electronics*, vol. 9, no. 1, p. 97, 2020.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9921826, 17 pages
https://doi.org/10.1155/2021/9921826

Research Article

A Survey on the Noncooperative Environment in Smart Nodes-Based Ad Hoc Networks: Motivations and Solutions

Muhammad Altaf Khan,[1] **Moustafa M. Nasralla**,[2] **Muhammad Muneer Umar**,[1] **Zeeshan Iqbal**,[1] **Ghani Ur Rehman**,[3] **Muhammad Shahzad Sarfraz**,[4] **and Nikumani Choudhury**[5]

[1]*Institute of Computing, Kohat University of Science & Technology, Kohat 26000, Pakistan*
[2]*Department of Communications and Networks Engineering, Prince Sultan University, Riyadh, Saudi Arabia*
[3]*Department of Computer Science & Bioinformatics, Khushal Khan Khattak University, Karak, Pakistan*
[4]*Department of Computer Science, National University of Computer and Emerging Sciences, Chiniot Faisalabad Campus, Chiniot 35400, Islamabad, Pakistan*
[5]*Department of Computer Science & Information System, Birla Institute of Technology & Science, Hyderabad, India*

Correspondence should be addressed to Moustafa M. Nasralla; mnasralla@psu.edu.sa

Received 17 March 2021; Accepted 22 May 2021; Published 30 May 2021

Academic Editor: Helena Rifà-Pous

In ad hoc networks, the communication is usually made through multiple hops by establishing an environment of cooperation and coordination among self-operated nodes. Such nodes typically operate with a set of finite and scarce energy, processing, bandwidth, and storage resources. Due to the cooperative environment in such networks, nodes may consume additional resources by giving relaying services to other nodes. This aspect in such networks coined the situation of noncooperative behavior by some or all the nodes. Moreover, nodes sometimes do not cooperate with others due to their social likeness or their mobility. Noncooperative or selfish nodes can last for a longer time by preserving their resources for their own operations. However, such nodes can degrade the network's overall performance in terms of lower data gathering and information exchange rates, unbalanced work distribution, and higher end-to-end delays. This work surveys the main roots for motivating nodes to adapt selfish behavior and the solutions for handling such nodes. Different schemes are introduced to handle selfish nodes in wireless ad hoc networks. Various types of routing techniques have been introduced to target different types of ad hoc networks having support for keeping misbehaving or selfish nodes. The major solutions for such scenarios can be trust-, punishment-, and stimulation-based mechanisms. Some key protocols are simulated and analyzed for getting their performance metrics to compare their effectiveness.

1. Introduction

Unlike other traditional data communication networks, ad hoc networks are considered very ideal in scenarios where rapid deployment of a network is preferred. The typical variants of ad hoc networks can be Wireless Sensor Networks (WSNs) [1], Mobile Ad hoc Networks (MANETs), Delay Tolerant Networks (DTNs), and Vehicular Ad hoc Networks (VANETs) [2]. Due to the exceptional features of wirelessly connected devices in ad hoc networks, these networks can be used for a variety of drives like gathering environmental data [3] and controlling smart homes, cities, and industrial equipment [4]. Since the nodes in such networks are designed to be inexpensive and small in size, these nodes have to work with a limited set of resources like storage, battery, processing, and radio frequency power [5, 6].

Each node in ad hoc networks can operate without the existence of a central router and can be programmed to support multihop communication for inexpensive and speedy utilization. In multihop communication, a source node can connect another target node through a chain of various intermediate nodes. Each node, besides its fundamental functionalities, also offers relaying services to other nodes to develop a collective, cooperative environment throughout the network. In some wireless ad hoc networks, nodes are deployed randomly and may move in undecided directions [7]. Moreover, such nodes form a dynamic topological structure that allows them to learn their degrees and route information by periodic exchange of information. Therefore, each node is desired to adequately coordinate with its neighborhood for keeping the route information updated.

The ad hoc networks are unorganized and infrastructureless networks. The nodes in such types of networks perform many operations along with the routing functionalities. The term smart node refers to the nodes which perform their operations autonomously without any input from others. These nodes intelligently adopt some strategies according to their own needs or preferences. In most of the game-theoretic approaches, the nodes are considered to be smart during the data routing in the network. These nodes are programmed in such a way that they learn from the environment and intelligently take decisions of their own interests.

The devices' scarcity of various resources in ad hoc networks can be associated with the nature of their exertion and physical structure. In WSNs, the nodes' processing power and energy are always assumed to be very limited. In almost all the routing protocols, the energy consumption rate is very deeply tested. Many research proposals are purely targeting the energy efficiency in WSNs [8–13] and fog networks [14], while in DTNs, the size of storage queues is assumed to have a vital role. The limited storage of devices highly degrades the opportunistic nature of data communication in DTNs. Moreover, the nonavailability of the relay or target node also reduces the network performance. In VANETs, various parameters are considered as important and limited. Primarily the bandwidth, storage buffers, and energy are considered as vital in such networks. Moreover, the high rate of mobility is also a challenge in VANETs [15]. In most ad hoc networks, the nodes operate on restricted batteries, which bounds the lifetime of the nodes and the entire network. In some cases, the social likeness or dislikeness of smart devices also affects data communication and network efficiency.

Each node consumes its resources on its own operations and the collective objective of the entire network. To accomplish the overall objectives, the nodes must cooperate with one another during the information exchange and data transmission from a source to a target node. While keeping the aggregate interest, each node consumes an additional amount of energy and storage queue for giving relaying services to other source nodes. This additional resource consumption can lead to shortening the nodes' life and can degrade its own data transfer. A node can eradicate these issues by simply not cooperating with other nodes. Nodes having a noncooperative behavior can be called selfish nodes. Selfish nodes prefer to be entertained by other nodes but in return do not like to consume their resource for others. The study in [16] defines two categories of selfish nodes: The first class of nodes participates in the routing by receiving and acknowledging the reception of packets. However, these nodes drop the received packets and do not forward any data or control packets generated by a source node. The second class of nodes does not take part in the routing and never accepts any route request packets (RREQ). Some authors also refer to selfish nodes as malicious nodes in their work [17]. In many literatures, the selfish nodes are considered nondestructive or nonmalicious as they only try to preserve their own resources. However, such nodes can influence the performance of other nodes and degrade the overall functionality of the network.

A node can adopt selfish behavior for its own advantage due to various reasons. In return, a selfish node can highly degrade the network performance up to a very high peak. The main issues caused by the existence of selfish nodes can be higher packet drops, increased energy consumption, imbalanced load among nodes, and nonavailability of optimal paths for data transmission. Additionally, a selfish node can be used by some malicious nodes for black hole attacks. In a black hole attack, the packets are intentionally dropped by the attacker nodes for a malicious purpose [18].

Various techniques have been introduced to manage selfish nodes in ad hoc networks. Some authors suggest the act of selfishness as beneficial up to some extent [19]. However, there should be a proper mechanism to control or allow such behavioral aspects of network nodes. The most popular techniques for selfish node management are trust management, incentive-based, and evolutionary game-theoretic mechanisms. In some cases, Intrusion Detection Systems (IDSs) can also be utilized to block noncooperative nodes in a network. Usually, the researchers propose their mechanisms to target some particular types of ad hoc networks. These mechanisms can be interchangeably considered for other types of the network with some modifications and assumptions.

The main object of this work is to enlighten the noteworthy aspect of the noncooperative environment caused by intelligent nodes in ad hoc networks. The primary drives for pushing smart nodes to act selfishly in different flavors of ad hoc networks are discussed and classified. Different domains causing the noncooperative communication environment targeted by various articles are discussed in this work. Moreover, several types of protocols designed for ad hoc networks and having support for selfishness management are studied and categorized. At the end of this work, protocols of different categories are simulated and their performance metrics are calculated. The differences and similarities of experimented protocols are discussed in detail and some valuable conclusions are made.

The article is divided into six sections. In the next section, a detailed description of the noncooperative environment in ad hoc networks is given. In this section, the fundamentals about the selfishness of nodes are discussed. Nodes can adapt selfish behavior due to various motives based on their preferences and limitations. A detailed description of the motivations for the adaption of selfishness is given in the third section of this article. The solutions for handling the selfish behavior of nodes are given in the fourth section. A selfish node can be isolated or/and stimulated by adopting some state-of-the-art schemes. In the fifth section, an analytical study is performed to check the effectiveness of various proposed techniques. In the last section conclusion of our work is given.

2. Noncooperative Behavior in Ad Hoc Networks

The routing protocols used in ad hoc networks can be categorized as proactive and reactive. In the proactive routing protocols, the nodes learn about the topology of the network and the availability of routes by exchanging some periodic messages. These messages are referred to as topology control messages or HELLO messages. Each node generates such messages after a particular period of time. Upon reception of such messages, the node responds with its availability. The generator and the respondent nodes keep their routing tables updated with the help of topology control messages. Destination Sequence Distance Vector (DSDV) and Optimized Link State Routing (OLSR) are the examples of proactive routing protocols. In reactive routing protocols, the message initiating node broadcasts a route request to all the network nodes to discover the network topology. The route discovery is made up by flooding the route request message throughout the entire network. These protocols are known as on-demand routing protocols. The destination node upon receiving the route request responds to the source node and a path is established between the two nodes. Examples of reactive protocols are Dynamic Source Routing (DSR) and Ad hoc On-demand Distance Vector (AODV) [20]. In both types of routing protocols the coordination and cooperation of nodes are always required.

The cooperative operation of the network nodes is always desired in ad hoc networks. In ad hoc networks node can be assumed as members of a community in which each member inputs something for the fulfillment of the combined objectives of the community. The contribution to problem solution binds the individual resource consumption with the aggregate interests of community members. However, nodes controlled by humans or programmed with intelligence may lead to some undesired circumstances for having self-interests. The nodes sometimes cannot judge the importance of their existence in the community. This self-interest can lead to noncooperation among the nodes [21]. If the developed infrastructure does not log the nodes' data traffic in an appropriate manner and allows nodes to smartly adapt their strategies, this leads to nodes' independence. An independent node may think that its own resources being meant for its use may lead to selfish behavior.

The selfishness of a node is somehow similar to a black hole attack in such networks. In the black hole attacks, the attackers intend to drop the data packets whenever they are supposed to forward those packets. The main intention for pack dropping in black hole attack is to degrade the network performance or another malicious purpose [22]. However, in the case of selfishness, the nodes adopt a noncooperative behavior only for their own benefit rather than any malicious objective. The existence of selfish nodes in such type of networks can lead to various unwanted outcomes. A selfish node can increase the number of overall packet drops but not giving a relay service to some or all source nodes. The selfish node can drop the received packets any time which are needed to be forwarded to the next hop in the network towards the destination. Due to this behavior, the load distribution is imbalanced among the network nodes. Consider a senior shown in Figure 1; a single selfish node can involve other normal nodes to become intermediate hops in a data transmission channel, causing the involved nodes to take the load which is not supposed to be taken. Due to imbalanced load distribution, the energy consumption is also not uniform. Some nodes exhaust their energies earlier and die before a normal period. The selfish nodes can increase the end-to-end delays by not allowing a source node to get the shortest paths towards the destination.

The article in [23] takes Dynamic Source Routing (DSR) as a case study and determines that the selfishness associated with routing can be classified into three major categories:

(a) *Type 1 Selfish Nodes*. The selfish nodes participate in the normal control data packets' transmission during the routing discovery and maintenance phases but do not become a relay for forwarding normal data packets. Such type of nodes is considered very dangerous for the overall operations in the data routing. These nodes initially participate in the route discovery to establish paths but later they start denying the relay service for others. In such cases, the packet drops and end-to-end delays are highly increased. It is also possible that the selfish nodes do not adopt noncooperative response for all the nodes but only selected nodes are targeted. The major reason can be social likeness or dislikeness.

(b) *Type 2 Selfish Nodes*. The selfish nodes do not participate in anything associated with data transmission for other nodes either in the route discovery phase or in the route maintenance phase. Such nodes only use their energies to be consumed by their own data processing and transmission. The routing protocols usually do not consider such types of nodes. No route information is gathered from or transferred to this class of selfish nodes. These nodes can highly degrade the overall data communication traffic and network connectivity. However, the routing protocols do not consider these as a major threat to the discovery and maintenance of routes in an ad hoc network.

(c) *Type 3 Selfish Nodes*. Such nodes adjust their cooperation level according to their resource levels. These nodes in the beginning act like normal nodes. With the passage of time, the nodes start declination in their cooperation with others due to a reduction in their resource levels. In a smart environment, it is possible that nodes interrelate their remaining energy levels with their selfishness levels. The multiple levels of selfishness, referred to as Multiple Threshold

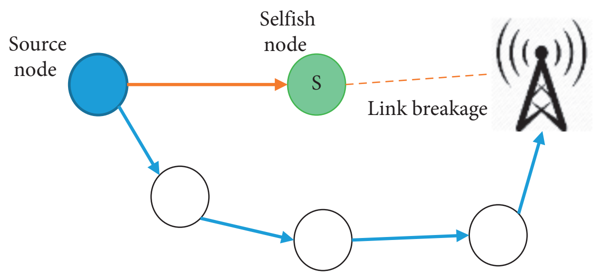


Figure 1: Effect of a selfish node in routing.

Selfishness (MTS), are well defined by [18, 24]. Such nodes are similarly dangerous as type 1 selfish nodes. The nodes support the route discovery flow for forming a topology but later interrupt the data flow by dropping data packets. Such type of nodes causes the routing protocol to reinitiate the route discovery process or adopt another alternate route for data transmission.

The selfish nodes usually adopt their distinct behavior for their own interest and do not develop an intention for the degradation of the network performance. However, it is understood that the selfish nodes bring unwanted and rapid topological changes in the network which put a very big impact on the overall performance of the network. Some authors, like Umar et al. [18], use the selfishness of nodes as a key for the establishment of paths and load balancing among the sensor nodes in a WSN. Some authors suggest that selfishness can be used in a positive sense and the network can benefit from such behavior of nodes up to some extent [19]. A mechanism allowing selfishness can permit some nodes to keep their resource preserved for some vital motives in the future. The nodes having some extra responsibilities can be permitted to not give any relay service to others in a selfish manner.

A node having persistent noncooperative behavior can raise the rate of packet loss in an ad hoc network up to 100%. However, the ratio of packet loss due to a selfish node diminishes with an evolution in the density of normal cooperative nodes in the ad hoc network [24]. Due to this assumption, we can say that the network having a large number of nodes will face comparatively less damage due to the presence of selfish nodes in it. More precisely, the number of selfish nodes can be directly associated with the network performance.

3. Motivations for Selfishness Adoption

A node in any variant of ad hoc networks can adopt selfishness whenever it is programmed with smartness or controlled by another intelligent entity. The intelligent entity can be another device, module, program, or human. Selfishness is very common in human-operated personal devices like smartphones and personal digital assistants [25]. A node can be adequately programmed so that it becomes independent of other nodes in the network and decides its own functionalities [26]. A smart node can adjust its behavior for several reasons. The foremost reason is the energy preservation in all the ad hoc networks. In Table 1, some ad hoc networks are given along with the potential motivations for their nodes' being selfish. All the motivational factors for pushing smart nodes to change their cooperative behavior are as follows.

3.1. Energy.

3.1. Energy. Almost in all types of wireless ad hoc networks, the nodes use batteries as the only power source. These batteries are usually disposable and non-rechargeable in most of the WSNs and similar networks. Nodes are randomly thrown into a field and then left unattended in such networks. Therefore, battery replacement or recharge cannot be made feasible, while in some cases these batteries can be recharged by the operators like in VANETs and MANETs. However, the energy source in ad hoc networks is always assumed as finite and scarce [11].

A node consumes its energy on various types of functionalities. The foremost are data processing, transmission, sensing, and reception. Energy consumption can be categorized as useful or wasteful. Energy consumed on data transmission and reception, data processing, and control messaging in the network can be considered as useful, while wasteful energy expenditure is carried by overhearing, generation, and processing of control packets, idle listening, and retransmission of lost packets [9]. If a node stops or reduces any of these functionalities it can be a detriment in various ways. To reduce the energy consumption on data transmission, it is possible for nodes to not forward others' data packets. The forwarding nodes usually consume additional energy on the reception of data and then retransmission. The selfish nodes reduce their energy consumption by never opting to give any relay service.

Suppose a node depletes ax amount of energy on transmitting a single bit and consumes bx amount of energy on the reception of a single bit. As per wireless communication fundamentals, the value of a must be greater than b, i.e., a>b [10]. Ignoring all other parameters used in data communication, i.e., range, etc., a normal node will consume $ax + bx$ amount of energy for forwarding a single bit, while a selfish node can save this by only hearing the single bit with bx amount of energy and does not use amount on retransmitting the same bit.

The selfishness of nodes for the sake of energy preservation is very common in most of the ad hoc networks. Most of the researchers take sensor nodes with disposable batteries as their case studies [18, 27]. However, any type of ad hoc network where the nodes are operated by a finite power source can be assumed to have the possibility of a noncooperative environment among the nodes. Some authors marked the selfishness of nodes as the most effective tool for their energy-saving and life-lengthening [18].

3.2. Storage Buffer. The nodes in ad hoc networks operate on limited storage space. It is obvious that the nodes must store the received contents temporarily before their transmission to the next hops. Sometimes the nodes need to choose which data content is useful for them and should be stored in their storage buffers. Most of the nodes try to keep their storage

Table 1: Types of ad hoc networks and their motivations for the selfishness adoption.

Network type	Primary concerns	Secondary concerns	No concerns
WSN	The nodes operate on limited energy, storage, and bandwidth.	Social likeness can be considered secondarily. Some intelligent nodes may prefer some other selected nodes for cooperation.	In most of WSNs, the nodes do not move. The mobility does not affect the behavior of nodes in WSNs.
DTN	Social likeness associated with limited storage buffer can be considered. The nodes prefer to use their storage buffer for some known nodes.	Energy can also be considered for behavioral change. The nodes may operate on a limited set of energy.	Privacy and mobility are not considered in most of the DTNs.
VANET	In most of the VANETs, the data privacy, social likeness, and mobility of network nodes are considered.	Bandwidth in some cases may be considered due to the huge amount of traffic.	Usually, energy and storage are not focused in most of the VANETs.
Smart phones ad hoc network	The smart nodes use a limited set of batteries. The phone may carry some sensitive private data and the users may have some social affiliations	The phone set may have a limited amount of storage for keeping the data being forwarded to others.	Mobility and bandwidth are not considered in such networks.
Mobile sensor network	Limited energy, storage, and mobility of nodes may affect the routing behavior.	Bandwidth, secondarily, may lead to a selfish behavior.	Usually, such nodes do not consider the data privacy.

buffer for their own collected data, which in return does not allow another node to be entertained. All the nodes keep the received relayed data saved with them until they get the connectivity with their next hops. It is possible that the nodes delete the received data before it is transmitted to sparing their storage buffer for their own data [28]. The limited buffer size leads to defining appropriate buffer management schemes. In some ad hoc networks, like DTNs, the buffer management policy describes which message to keep and which to discard. This policy of prioritizing messages for data buffer is assumed to be very fruitful by [29]. In DTN, the nodes can adopt a selfish behavior due to their limited buffer storage which in turn can benefit them for their own data transmission. However, the source nodes needing multiple hops towards their destinations are highly affected. The selfish nodes in such case never bother to request others for relay service with buffer usage for their data transmission but in return regret for not giving any space in their buffer to keep the relay requesters data messages. Such act of selfish nodes is also referred to as routing misbehavior [28]. Figure 2 shows some nodes using their limited storage buffers during data communication.

3.3. Social Likeness. Internet-based social networks have made many interconnected relationships among the users. Such networks touch countless aspects of our daily lives and enable us to get people having similar mindsets. In social networks, the selfishness of the user cannot be overlooked in many terms. The far most reason for selfishness is the likeness and dislikeness [30]. Similarly, the smart or humanly controlled nodes in an ad hoc network can also consider the aspect of the social likeness during their communication. Practically, sometimes the intermediate nodes drop the packets due to not giving any attention to the sender node. Social selfishness is usually associated with the DTNs type of networks in which most of the nodes are either operated by humans or installed in vehicles [31]. The nodes having no social ties do not cooperate with one another with a determination to save their resources. An example of the mobile social network can be considered for such a case. People like to share common interests which form a community via mobile phones. These communities can be considered as interest groups of mobile nodes where each member tends to assist its community members only and does not like to spare its resources on any stranger node. Moreover, previous connectivity and behavioral records can also influence a node to socially like or dislike others [32].

3.4. Bandwidth. In a network where nodes transmit bulky data their assigned bandwidth may need to be utilized. The size of bandwidth is always limited in nature. It is possible that the nodes adopt particular cooperation or noncooperative behavior according to the size of their assigned bandwidth. Sometimes, the nodes cannot entertain any other relay requester for data transmission due to the own bulky data [31]. Bandwidth limitation in most of the ad hoc networks is a considerable issue and addressed by many researches. In WSNs, the sensor nodes operate on a very limited bandwidth as they cannot manage a higher data spectrum with their lower energies and processing capabilities. Many authors proposed node-level processing to reduce the load on communication bandwidth [33]. Therefore, the nodes, if programmed with intelligence, prefer to use their communication channel for their own data.

3.5. Mobility Rate. In most of the MANETS, due to the mobility of nodes, the topological interconnections change with the passage of time. If a node moves from one place to another, it may break some connections and may establish some new connections. It is palpable that the higher rate of mobility nodes can exceedingly degrade the network performance. Sometimes, a relay node cannot change its location in the network to avoid any data loss of the source node. It is also possible that a source node requests the relay nodes to not move until the completion of the data

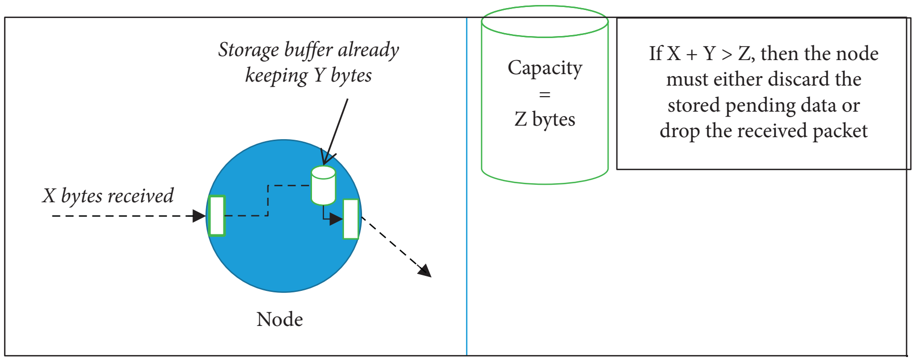

Storage buffer already keeping Y bytes
X bytes received
Node
Capacity = Z bytes
If X + Y > Z, then the node must either discard the stored pending data or drop the received packet

Figure 2: Use of storage buffer during data transmission.

transmission. Since each smart node considers its own benefits, they may act selfishly and change their location without favoring any source node. Moreover, mobility control can be incorporated into the mechanism for fault tolerance [34]. The concept of cooperative mobility and communication appeared in the related literature in the 1970s. This key meaning is that each node in the network has two possible modes, i.e., (a) selfish, which does not move for others but only prefers its own optimization, and (b) cooperative, which takes care of the entire network and tries to attain the aggregate goal [35].

3.6. Privacy Concern. During communicating with each other, some nodes may require others' private data. In a general context, we can say that a network of mobile phones exists and each phone shares its location with others through some applications. It is possible that a phone reads the locations of all the connected ones but never likes to share its own location. Such type of selfishness can be categorized into privacy or personal data sharing. According to a scheme proposed by HE et al. [36] for WSNs, clusters may share common data with each other while inside clusters some nodes may need to share their private data.

4. Solutions for Selfishness Management

Many researchers target a primary domain like energy efficiency, load balancing, reduced end-to-end delays, and efficient storage buffer management while designing a selfish node management protocol. Various classifications of selfish node management schemes have been proposed in the area of security and routing in ad hoc networks. The schemes can be classified in various ways. In this work, we divide the selfish node management schemes into four classes: (a) IDS; (b)trust-based mechanism; (c) incentive-based mechanism; (d) evolution games theoretic approaches. Each class target some particular types of ad hoc networks and have some pros and cons. For example, in WSNs, lightweight schemes which do not put extra load on radio transmission are preferred. Some articles like [18] suggest credit-incentives-based schemes as a more suitable solution handling non-cooperative environment in WSNs. The mentioned classes are explained in Table 2.

4.1. Intrusion Detection System. An IDS can be a device or an embedded program that monitors the data traffic in a network and detects any inappropriate behavior violating a predefined policy or pattern. A typical IDS provides information about the nodes' abnormal/malicious behavior in a network. However, the same can be utilized for the detection of selfish nodes in the network. The IDS can be used for detection purposes only and cannot be utilized for node stimulation towards cooperation [41]. There are three major classifications of IDSs: misuse-based detection, anomaly-based detection, and specification-based detection [22].

4.2. Trust Management Schemes. Some proposed systems use a trust development procedure among the nodes. The trust levels are defined based on the nodes' cooperation level in the network. The nodes share their experiences about one another which ultimately let all the nodes understand each other's behavior. The knowledge-sharing about the past experiences of nodes leads them to develop a trust level about each node in the network. The selection of relay relies on the trust level of the next nodes in the potential route [42]. Various typical and sophistical schemes have been introduced in this domain. Most of the schemes are considered as the fundamental and classical schemes for addressing selfishness in ad hoc networks. The trust is usually associated with the nodes; however, the same can also be applied with the data. Data trust can be used to verify the authenticity of data in the various types of ad hoc networks. Some authors give a description of node trust and data trust in the VANETs [43].

An old but still the most effective approach, watchdog and pathrater [37], is a selfish node detection and punishment scheme. This approach is used to detect routing faults by monitoring the behavioral aspects of involved network nodes. This scheme targets selfish and malicious nodes. Watchdog is a detection module while pathrater is used to block the misbehaving or problematic nodes. Another similar approach, CONFIDENT [44], targets the malicious nodes in a network. Four major modules are designed for this approach. These modules are programmed in each network node to achieve the aggregate goal of the optimal performance of the network. Many articles are using these two techniques as baselines for their proposed mechanisms.

Table 2: Classes of schemes used for selfish node management.

Class type	Description	Example
Detection mechanisms	Detects and adaptively reports/blocks the noncooperative or misbehaving nodes in the network Simple and straightforward schemes for treating all the nodes	Marti et al. [37]
Trust-based mechanisms	Some trust levels are defined among the nodes Behavioral history of each node is logged and the nodes consider the trust levels among them	Shaikh et al. [38]
Reputation-based incentive mechanisms	A reputation level is made based on incentives granted Nodes try to get more incentive by offering frequent relaying services for their better reputation	He et al. [39]
Credit-based incentive mechanisms	A pricing model is made Nodes are given some values for their data exchange The relaying service is paid by the source nodes for forwarding their data	Umar et al. [18]
Evolutionary game-theoretic approaches	A repetitive type of procedure is adopted in such mechanisms Each node learns with the passage of time and adjusts its strategies to obtain an equilibrium point for the entire network Most of the evolutionary games are applied in cluster-based WSNs	Gameda et al. [40]

Some trust-based mechanisms like [38] are effective in group or cluster-based ad hoc networks. The profiles of all the nodes are kept and distributed among all the nodes through the group heads. The trust values are directly affected by the behavioral aspects of nodes during the communication and cooperation among the nodes. For effective and secure routing some trust-based schemes also incorporate public keys for trust maintenance.

The protocols laying in this class can also be referred to as node punishment-based mechanisms. The ultimate goal of carrying the detection and trust management in such schemes is to punish the noncooperative or misbehaving nodes by blocking them in the network. The blocked nodes are also called black-listed nodes. Such nodes are not entertained for their own relay request by other nodes. Moreover, these nodes are not requested for any cooperation during the data transfer by any source node.

The trust-based schemes are also introduced in the emerging M2M and IoT architecture. In such types of ad hoc networks, the heterogeneity of nodes is also considered at the base level [45].

4.3. Incentive-Based Mechanism. Many researchers assume the incentive-based mechanisms as the most effective techniques in the same domain. These approaches consider the network nodes as rational and autonomous of any restriction for cooperation with one another. The nodes are supposed to have their policy for adopting a work contribution strategy based on their individual interest. In such scenarios, the nodes are stimulated by using some incentives to let them cooperate with each other. The theme of these approaches can be simply called "give and take" or "tit-for-tat" procedures. The incentive-based mechanisms can be classified as reputation-based incentives and credit-based incentives. For designing such systems, many researches proposed the incorporation of game theory in their proposed mechanisms. The role of game theory and the two classes of incentive-based mechanisms are explained in the following subsections.

4.3.1. Role of Game Theory. For the development of incentive-based mechanisms in ad hoc networks, game theory is given a vital role in the realization of the procedural analysis and quantization in the designing phase. Basically, the game theory is introduced in economics for the evaluation and processing of financial matters. This area is also used in social and biological sciences. However, this theory is recently adopted by many researchers for their proposals in wireless networks. A game can be considered as a set of players, strategies of each player, payoff functions, the output or gains, and the equilibrium function. A typical ad hoc wireless network can be aligned with the game theory by taking network nodes as players, strategies as features and actions of nodes, and the payoff functions as the point where a node can balance its work with its energy consumption efficiently. The output can be considered as the outcome in terms of various concerns in a network like energy efficiency, bandwidth usage, nodes' storage usage, and overhead on each node. Finally, the equilibrium point in a wireless network can be considered as a situation in which each node gets its optimal position addressing the aggregate benefits of the entire network.

In most of the incentive-based selfish node management systems, game theory is used to intelligently handle the nodes' behavior according to the needs of a network. Various game types can be incorporated into the incentives schemes. Examples of games are cooperative, noncooperative, repetitive, evolutionary, and bargaining games. The selection of a game type for a scheme is dependent on the nature of the network and the requirements. For example, repetitive games are suitable for such networks where nodes do not keep all the information in the beginning. Similarly, evolutionary games are considered more effective in cluster-based ad hoc networks.

4.3.2. Reputation-Based Mechanisms. The main theme of such techniques is inspired from the usage of reputation levels of users in web-based services like Amazon and eBay. The sellers and buyers are assigned some points according to their behavior. The nature of users in such web-based stores can be judged by looking into their earned points. A similar concept can be used to evaluate the nodes' participative behavior in ad hoc networks. In the reputation-based mechanisms, several reputation stages are made to classify the nodes according to their level of participation. Those nodes which do not cooperate or have less than a specified level of cooperation are punished by other nodes. Usually, such nodes are not offered any relay service by others. To obtain an acceptable state of behavior, each node tries to obtain adequate incentives by adaptively offering its services to other nodes. It is a kind of stimulation in which each node is pushed to cooperate for the sake of its better reputation in the network [39].

The trust and reputation of network nodes cannot be considered similar due to many reasons. The trust is an active entity while reputation can be considered passive. Trust is a kind of peer node's belief and so can be extended from a peer to its node, while reputation is the perception level of nodes about each other. In some articles the trust is also associated with risk factors and reputation is something based on the history of a node.

Paper [46] points out the major issues associated with the reputation-based incentives mechanisms. The foremost issue is that these approaches do not adequately handle the node assessment process. The second issue is that the structuring of groups of nodes cannot be designed efficiently in ad hoc networks. The third issue with reputation-based mechanisms is that the nodes used their radio transmission excessively for obtaining information about each other. Therefore, many authors suggest the usage of credit-incentive-based mechanisms for controlling the nodes' behavior in a network.

4.3.3. Credit-Based Mechanisms. These schemes are also called pricing-based schemes. Such schemes consider the data transmission and relaying support by network nodes as a service that must be paid. In credit-based incentive schemes, the worth for handling buy and sell or lease and rent mechanism is obtained by several forms of values. Some of these are referred to as virtual currency, scores, money, and points. The pricing schemes also need an additional log for keeping the record of exchanges of these values. Each node upon giving relaying service obtains an amount of virtual currency from the source nodes. Each relaying node earns this currency and uses it for its data transfer. Nodes having no or fewer values of currency cannot be able to pay the relaying service and so cannot transmit their data. In such a scenario, each node tries to maintain a trade-off between its resource consumption and virtual currency collections. Such techniques give nodes a degree of intelligence for optimization of their resources along with the network performance [46].

The credit-incentive schemes can be applied in many ways. Some articles introduced a bargaining environment between the relaying and the source nodes. The game theory of a bargaining model is used in such schemes. The main purpose is to make a competitive environment among the network nodes. Moreover, the calculation of currency for buying and selling is also aligned with some procedures. For example, the key parameters of nodes, i.e., energy, storage, and network hierarchical level, etc., can be considered for fixing the amount of currency.

In some ad hoc networks, like WSNs, nodes are connected to a central control through some hop nodes. Some nodes are directly connected to the central station and do not need any relaying service. Therefore, such nodes do not need to cooperate in any way and do not care about any currency maintenance. To overcome such cases, the central control also takes some exceptional measures by applying a reputation or punishment-based mechanism [18]. Some authors also proposed a movement cost for letting the nodes move for the sake of an aggregate benefit [34]. The nodes are given some benefits for their sacrificial movement in the field in a cooperative manner. The main purpose of such incentives is to stimulate the network nodes for moving in the area for the sake of fault tolerance.

4.4. Evolutionary Games. In evolutionary games, the nodes primarily do not use their strategic reasoning in the initial stage. All the nodes in the network learn from their experiences and then develop a model to design their strategies. This game is also known as a repetitive model in which the network nodes learn with the passage of time [40]. In the same manner with the passage of time the nodes evolve their behavior and adjust the cooperation at an optimal level where both the individual node and the entire network are benefitted. The most stable situation in such type of mechanism is referred to as an evolutionary stable strategy or evolutionary equilibrium. Most of the evolutionary-based mechanisms are designed for cluster-based WSNs [8, 40]. The evolutionary games can also be incorporated into the trust-based and incentive-based mechanism. Table 3 shows some proposed schemes for handling selfish nodes in ad hoc networks.

Table 4 shows some classical mechanisms which are considered useful as guidelines and base techniques for developing new schemes. Most of these are taken as baselines for the development of advanced techniques for selfishness management in ad hoc networks.

5. Analysis of Proposed Schemes and Discussion

In this work, a comparative analysis of five different protocols is made by taking a WSN as a model network. These protocols are DSR[ref], DSR with selfish nodes [18], Reward-based Mechanism (RwBM) [18], GREET [40], and GTMS [38]. The results are made by varying three major parameters, i.e., the number of nodes, pause time, and the ratio of selfish nodes. The results are calculated for taking the five performance metrics in the network, i.e., energy consumption, end-to-end delays, throughput, packet delay ratio (PDR), and packet loss ratio. The work is simulated in NS2.35 under Ubuntu operating system. The key parameters for simulation are listed in Table 5.

Table 3: Proposed schemes for selfish node management in ad hoc networks.

Article	Mechanism	Description
Attiah et al. [8], 2018	Evolutionary game	(i) A route selection problem is modeled by using a game-theoretic approach. (ii) The replicator dynamics mechanism is used to indicate that the nodes can be trained from their strategies and hence modify their strategies sets with time.
Subba et al. [41], 2018	Detection	(i) The work combines a lightweight neural network with specification rules for anomaly detection to identify misbehaving nodes in the network. (ii) A Bayesian game is designed which takes the IDS and the sensor nodes as two noncooperative players.
Umar et al. [18], 2018	Incentive-based	(i) Virtual currency referred to as scores is used for selfish node stimulation. (ii) Scores are calculated by taking many nodes' and network's parameters.
Raja et al. [42], 2018	Trust management	(i) The selection of cluster heads is done by using the multiple constraint aware glow worm swarm optimization approach (MC-GSO). Every node is evaluated in the network according to this approach. (ii) Various objectives are achieved for the trust metrics during the cluster head selection.
Yang et al. [47], 2017	Incentive-based	(i) Particularly designed for clustered WSN with an aim to balance the consumption of energy among all the nodes. (ii) A convex payoff function is designed for the behavior of each node. The game is derived with the help of this convex optimization.
Gemeda et al. [40], 2017	Evolutionary game-based	(i) Targets cluster-based WSNs. (ii) Mainly focus on the cluster heads manipulation and selection process.
Yu [48], 2016	Incentive-based	(i) Nodes are pushed to cooperate in routing by using some incentives. (ii) Some values are exchanged for getting data communication and relay services.
Li et al. [43], 2015	Trust management	(i) The work is proposed for VANETs where the trustworthiness of both mobile sensor nodes and transmitted data is evaluated. (ii) The recommendation trust and functional trust are categorized to indicate nodes' performance.
Duan et al. [17], 2014	Trust management	(i) A trust-aware routing frame is designed by incorporating lightweight and proficient attacks resistant mechanism. (ii) The common features associated with attacks in terms of trust awareness are addressed. (iii) The trust derivation is based on the analysis of results.
Chen et al. [21], 2013	Detection	(i) The cooperation level of each node is calculated and the selfish nodes are punished by blocking them in the network.
Xu and Guo [49], 2012	Incentive-based	(i) Particularly target opportunistic networks. (ii) The incentive exchange is made through various rounds of a bargaining game.
Bao et al. [50], 2012	Detection	(i) Uses highly expandable cluster and hierarchical trust-based management protocol for efficiently detecting the malicious and the selfish.

Table 4: Fundamental/classical selfishness management schemes.

Article	Mechanism	Description
Marti et al. [37], 2000	Reputation and detection	(i) Well-known as watchdog and pathrater (ii) Watchdog detects misbehaving nodes and pathrater blocks targeted nodes
Boudec and le [44], 2002	Trust and detection	(i) CONFIDENT, a reactive routing protocol that uses four major modules for detection and blockage of selfish nodes
Buttyan and hubaux [51], 2003	Detection and incentive-based	(i) Only the cooperative nodes are allowed to transfer their data (ii) The concept of virtual currency is used by introducing packet trade and packet purs
Zhong et al. [52], 2003	Incentive-based	(i) A credit-based incentive exchange scheme is used (ii) The scheme particularly targets mobility in MANETs like networks
Chen et al. [26], 2011	Evolutionary game-based	(i) Nodes operate according to a predefined set of states (ii) The nodes adjust their selfishness level with time

5.1. Energy Consumption. Since energy consumption is always considered in ad hoc networks, almost all the protocols designed for such networks are evaluated in terms of the nodes' life. The energy consumption rate can be affected by the simulation parameters and the routing protocol design. A network energy consumption can be evaluated by considering either the routing energy consumption or the average energy consumption. In routing energy consumption, the network layer of the protocol is checked only to determine the energies of nodes in a network, while the average energy consumption is the mean value of all nodes' consumed energies. Energy can be calculated by taking the

Table 5: Simulation parameters.

Parameter	Value
Area	500 × 500
Network type	WSN
Number of nodes	50–300
Ratio of selfish nodes	Up to 5%
Node distribution	Random
Comparisons	DSR, DSR with selfish nodes, GREET, GTMS, and RwBM
Initial energy	100
Rx power	0.3
Tx power	0.6
Size of the packet header	4 bytes
RSc header size (RwBM)	4 bytes
IBSc header size (RwBM)	4 bytes
Movement trace	Off
Cluster size (GREET)	Game-based (varying)
Cluster size (GTMS)	9 nodes
Traffic source	CBR
Packet protocol	TCP
Threshold distance for CNs (RwBM)	50
Threshold participation (RwBM)	0.4
Threshold lambda (RwBM)	0.5

sum of transmit, idle, receive, and sleep power in all layers in a simulation environment. The existence of selfish nodes in an ad hoc network can highly affect the rate of energy consumption in the overall network.

In Figure 3 the average energy consumed over 15 different time pauses of the experimented protocols is recorded. For this experiment, a set of 100 nodes with 5% selfish nodes are taken. The DSR protocol is mainly designed for ad hoc networks having mobile nodes. In WSN DSR does not utilize its features for handling the mobility of nodes. Therefore, its performance may be not optimal as compared to specialized protocols designed for WSNs having static nodes. The performance of SELFISH-DSR is the worst in the figure. It is because there is no such selfish node handling mechanism. GTMS is quite responsive by giving a moderate level of results for energy consumption. This mechanism mainly utilizes the trust reciprocity among the nodes. The technique is much better than DSR and SELFISH-DSR protocols. However, due to a simple mechanism for reputation and mutual trust mechanism, GTMS is giving comparatively lower results than GREET and RwBM protocols. GREET has the lowest level of energy consumption at most of the time. Since this protocol is cluster-based and mainly designed to evaluate all the previous strategies of nodes, it has put less communication overhead on nodes for passing control and information messages. RwBM initially loads the scoring mechanism which takes some time to give the accurate result values. In initial pause times, it is giving very low values which indicates that the communication is not started properly. GREET, compared with RwBM, gives comparatively the best results for many time pauses. However, due to its cluster-based type, after a longer period, the increase in energy consumption can be noted due to the rapid elimination of nodes from the network. In cluster-based networks, all the nodes may start dying rapidly one after another.

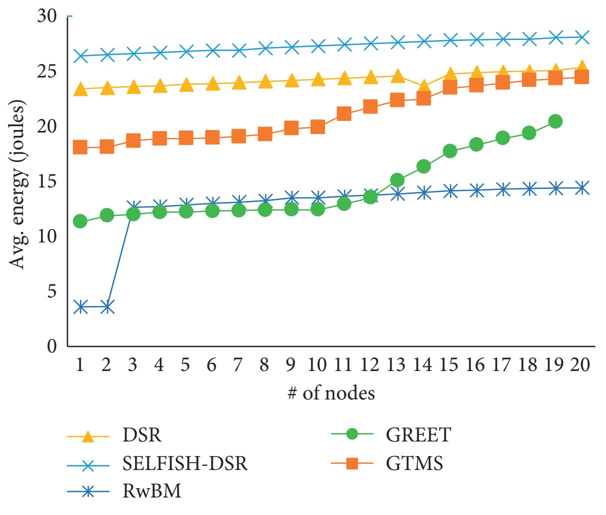


Figure 3: Average energy consumption at twenty time pauses.

In Figure 4, there are 5% selfish nodes and values are noted at time pause 10. All the mechanisms are giving consistent values except the SELFISH-DSR. Since there is no mechanism for the selfishness of nodes, energy consumption is increasing with the increased number of nodes. The number of selfish nodes is proportional to the number of nodes but the placement of selfish nodes over the same area greatly affects the performance of SELFISH-DSR. GTMS is giving similar results as in the previous experiment of pause times. However, its performance is declining with a number of nodes higher than 200. We can assume that the trust mechanism may start failing with a large number of nodes. It is also possible that the trust mechanism with densely deployed nodes may not be efficiently effective. The incentive-based mechanism, RwBM, is giving almost similar values for all the variations in the number of nodes. RwBM has a mechanism to deal with the nodes' individual

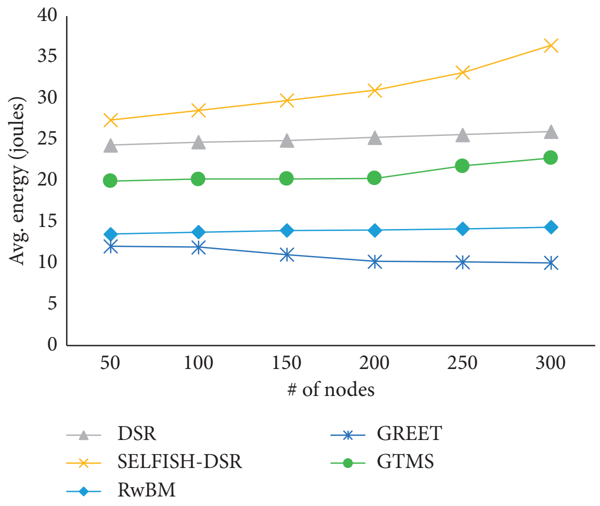


FIGURE 4: Average energy consumption with an increasing number of sensor nodes.

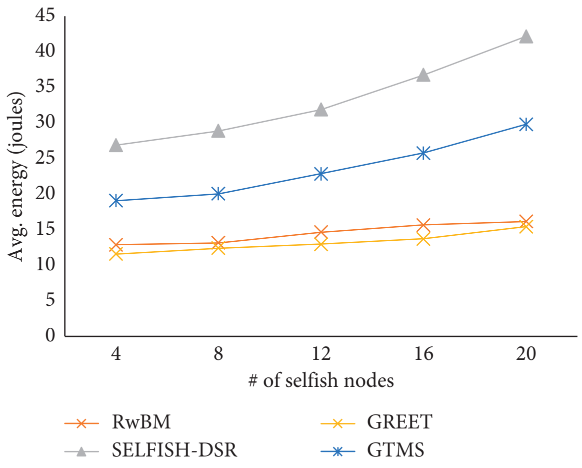


FIGURE 5: Average energy consumption with an increased number of selfish nodes.

importance based on the nodes' density in the network. Therefore, the mechanism aligns the values of incentives according to the number of nodes and the average energy consumption is not affected by the increased number of nodes, while GREET has very impressive results by giving the least values in this experiment. The energy consumption by nodes is decreased with the increased number of nodes. It is because the mechanism is based on evolutions and the nodes learn from each other and the time period. As the number of nodes is increased, the optimal point or the nodes' maturity level is obtained earlier, and the optimal strategies are adopted by all the nodes well in time.

Figure 5 shows the third experiment relating the average energy consumption of the nodes in a WSN. There are 100 sensor nodes and the time pause used for recording values is 10. The response for an increased number of selfish nodes is notable in all the protocols. SELFISH-DSR is giving support for selfishness; therefore, the energy consumption is highly increased with each rise in the number of selfish nodes. The performance of GTMS is also not well with the higher number of selfish nodes. It is because of the mutual reputation and trust mechanism. Each node mutual with a selfish node also changes its behavior. GREET and RwBM are giving similar results. These protocols have incorporated appropriate mechanisms for handling selfish nodes in the network; therefore, both are giving very little hike in the energy consumption against the increased number of selfish nodes.

5.2. Throughput. Throughput is the total quantity of data that reaches a destination node from the source node at a particular time. It can be used to determine the effectiveness of a routing scheme.

In Figure 6, the average throughput of SELFISH-DSR is very low at 6 kbps for almost all the pause times. It is giving lower throughput than a normal DSR in a network without any selfish node. The throughput of GTMS is moderate and consistent due to its communication style. The nodes are taken in a smooth flow and equally treated in GTMS. Therefore, the throughput of this protocol is slightly increasing with time and becomes consistent after time pause 9. The throughputs of RwBM and GREET are very unpredictable but higher than GTMS and DSR protocols. RwBM does not give any value at time pause 1 in our simulation environment due to its score loading and configuration process. Later due to scores' exchange and periodic updating of each node the throughput highly fluctuates. GREET initially has similar throughput as in RwBM. In GREET, the nodes learn repetitively by using an evolutionary game. Therefore, the throughput is lower and cannot be predicted at initial pause times. Later once the nodes learn from the environment, after time pause 5, the nodes adjust their utility functions and deliver a better throughput.

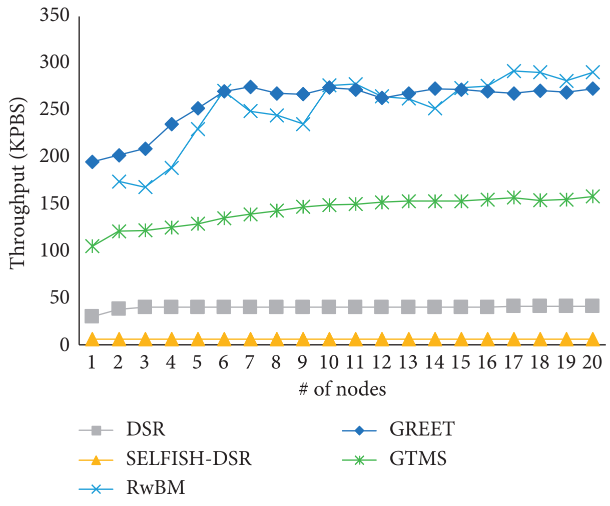


FIGURE 6: Average throughput at twenty time pauses.

The effect of a varying number of nodes on the experimented protocols is shown in Figure 7. The values are recorded at time pause 10 and the ratio of selfish nodes is 5%. The results are similar to the previous figure. The evolution game-based approach, GREET, is giving a very positive

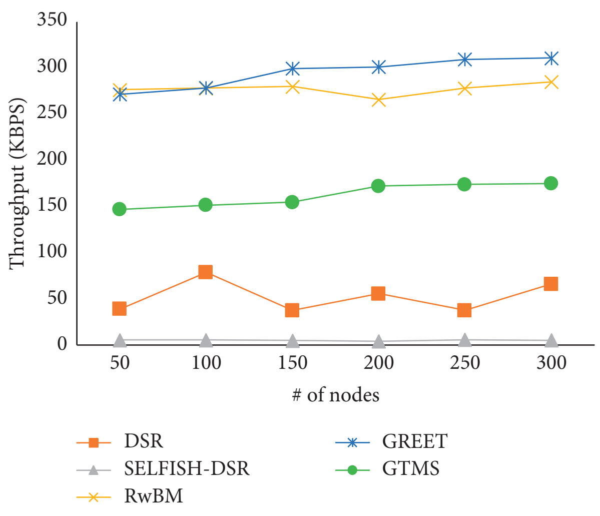


FIGURE 7: Average throughput with an increasing number of sensor nodes.

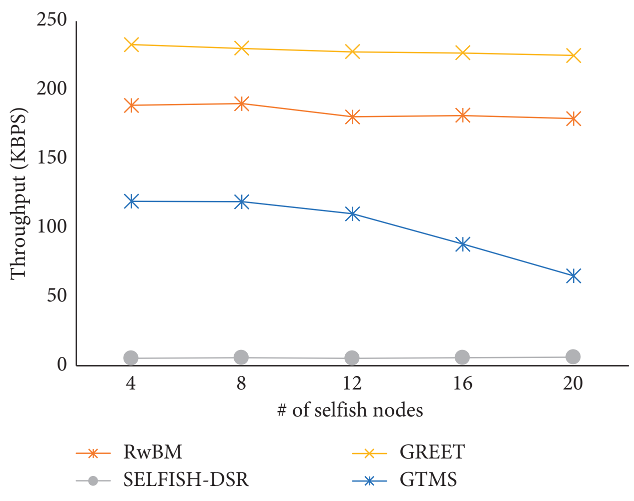


FIGURE 8: Average throughput with an increased number of selfish nodes.

response to the increased number of nodes. RwBM is giving similar fluctuating results for each experiment. Here again, we can conclude that the RwBM, due to its scoring mechanism, gives slightly random results. However, RwBM is giving much better results than GTMS and DSR. The throughput of GTMS is also increased in this experiment.

In the last experiment for throughput, shown in Figure 8, 100 nodes are taken. The performance of GTMS is very low with an increased number of selfish nodes. It cannot manage the higher number of selfish nodes that leads to a lower value of the throughput. RwBM and GREET are also marginally affected. However, this decrease in the throughput can be considered as a very minor effect. In this experiment, the GREET outperforms due to its advanced mechanism.

5.3. End-to-End Delays. In ad hoc networks, the high data rates are not essentially required but delay constraint is greatly considered. If the required information is delayed, then it might be not useful in the network. Therefore, we calculate the average delay rate in each protocol to determine their performance. The packet end-to-end delay is the meantime that a packet consumes to reach the destination from a source node. In our scenario of WSN, we are taking the base station as the destination node. Delays in a network usually associate the speed of MAC control exchange, buffer queues, radio transmission, and routing mechanisms. In our assessment, we are comparing the routing mechanism and all other parameters are considered as similar.

In Figure 9, nodes are taken. The ratio of selfish nodes is 5%. The experiments indicate that the end-to-end delays for DSR are higher than other protocols. DSR can be more efficient if used in a mobile node environment. Moreover, reactive protocols are less efficient than proactive protocols in delays matric. The reputation and trust levels in GTMS are developed over time. Each node considers the history of its connected hops; therefore, in the higher pause times, the delay is decreased. GREET initially produces higher delays due to its learning nature. The nodes cannot respond quickly

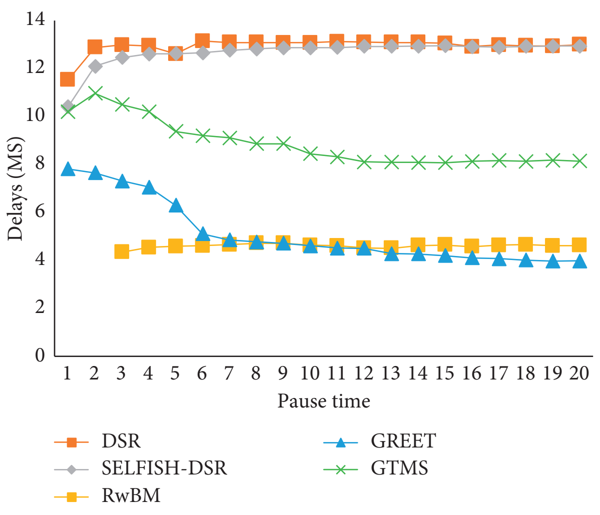


FIGURE 9: Average end-to-end delays at twenty time pauses.

and the act of selfishness is not handled properly. Moreover, in cluster scenarios it is common to have higher delays in the initial stages. RwBM and GREET are giving similar results after time pause 6. RwBM does not give the values in our simulation environment which is most likely due to the scoring mechanism and the adjustment of selfishness by each individual node. Moreover, the work is taking all the parameters of nodes which takes some time to be communicated and configured properly. One of the nodes is configured and the scoring mechanism is loaded; then the delays become similar at all the time intervals.

Figure 10 shows the end-to-end delays with respect to an increased number of nodes in a network. The results show that GREET is the only protocol that gives no effect to the delays with the changed number of nodes. RwBM is also giving somehow similar results. RwBM also uses a card system which may take some time to process. In the card system, some nodes are blocked. Node blockage can also increase the delays in a network by breaking some routes.

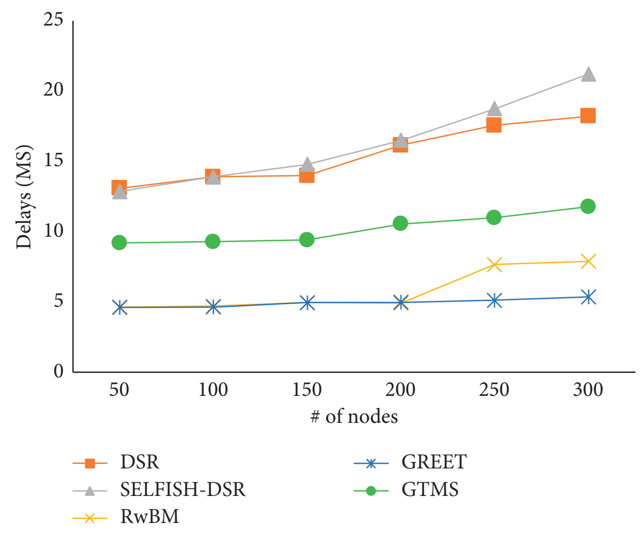


Figure 10: Average end-to-end delays with an increasing number of sensor nodes.

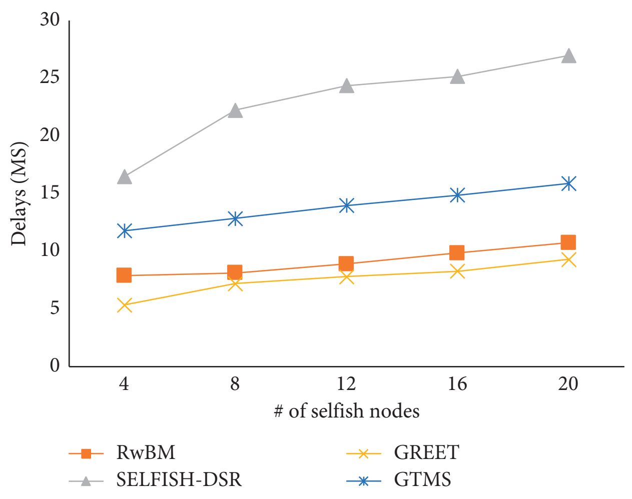


Figure 11: Average end-to-end delays with an increased number of selfish nodes.

However, the change is noted at 200 and 300 nodes in the experiments. GTMS is giving similar results till 150 nodes. It is also affected by the higher number of nodes in many experiments.

Figure 11 shows the end-to-end delays with an increased number of selfish nodes in the network. A set of 100 nodes is taken for these experiments. The response ratio of delays in all the protocols is similar. Each protocol is giving a negative result due to the increased number of selfish nodes. In comparison, GREET and RwBM are giving the best results in these experiments.

5.4. Packet Delivery Ratio. PDR is the ratio of the number of data packets sent by a source node to the number of data packets received by the source node. This metric can be used to measure the success or loss rate and characterizes the efficiency and correctness of routing protocols in ad hoc networks. The higher PDR indicates the better performance of a protocol.

Figure 12 shows the PDR for each node at different time pauses. 100 nodes with 5% selfish nodes are taken. Almost all the protocols except SELFISH-DSR are giving similar PDR. Initially, GREET gives lower PDR due to its learning phase. RwBM is giving appropriate values for PDR after time pause 4. SELFISH-DSR does not keep any selfishness management; therefore, its performance in this experiment is the worst.

Figure 13 shows the PDR for each protocol with the increased number of nodes in the network. Here again, the results of all the protocols are similar except SELFISH-DSR which has lower but consistent PDR for all the sets of nodes. As the number of nodes is increased, the PDR value is also increased. It is due to the availability of multiple routes and the decreasing possibility of packet losses. All the results are taken at time pause 10 and the ratio of selfish nodes is set to 5%.

The number of selfish nodes, if increased, also does not affect the experimented protocols as shown in Figure 14. The

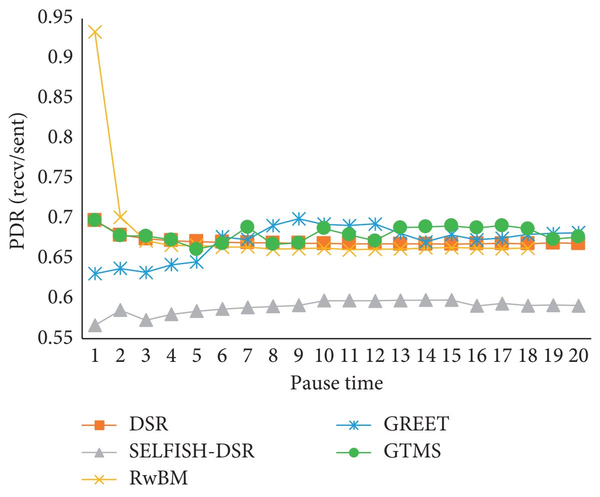


Figure 12: Packet delivery ratio at twenty time pauses.

PDR is decreasing in SELFISH-DSR only due to its incompleteness. In GREET, every node is a selfish node; therefore, the increased number of selfish nodes does not affect any change in its PDR value. RwBM, as usually, is giving uneven PDR for each number of selfish nodes. It is due to the provision for selfishness level adjustment and scoring and card system in the mechanism. GTMS is comparatively giving very interesting results by producing higher PDR values for the higher number of selfish nodes. Nodes operate on trust levels and selfishness is managed through reputations and trust reciprocation among the nodes.

5.5. Packet Loss Ratio. The packet loss ratio is used to get the failure rate of reception of transmitted packets. This value can be associated with signal degradation, the existence of misbehaving or selfish nodes, and routing mechanisms. In Figures 15, 16, and 17, the results are reflected by the results for PDR already discussed. Here in these experiments, the

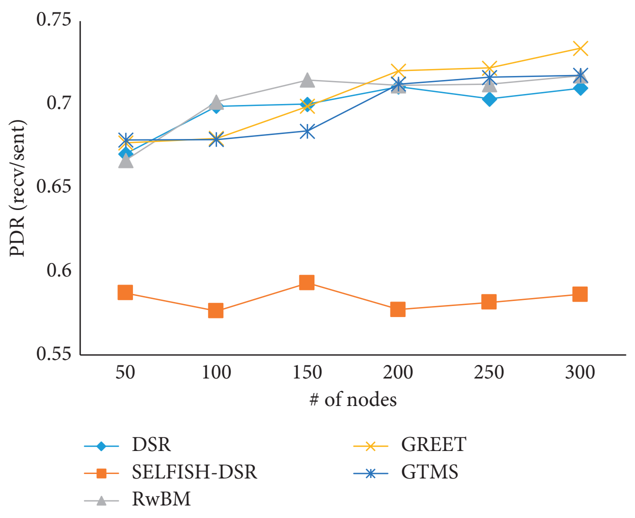


Figure 13: Packet delivery ratio with an increasing number of sensor nodes.

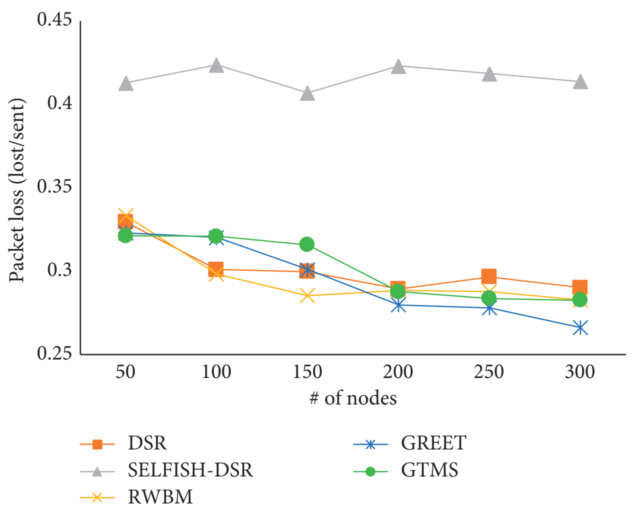


Figure 16: Packet loss ratio with an increasing number of sensor nodes.

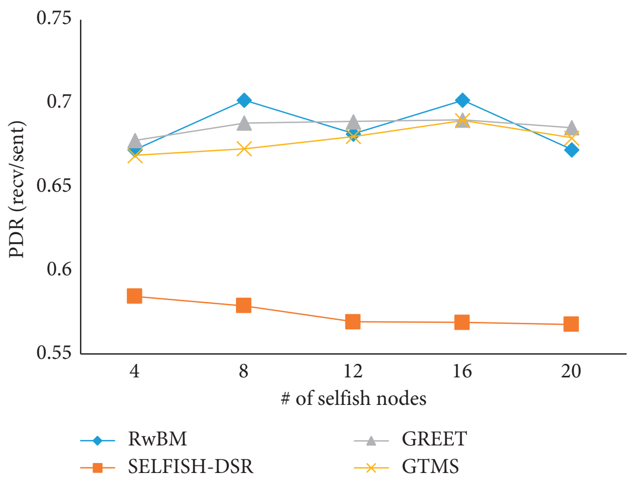


Figure 14: Packet delivery ratio with an increased number of selfish nodes.

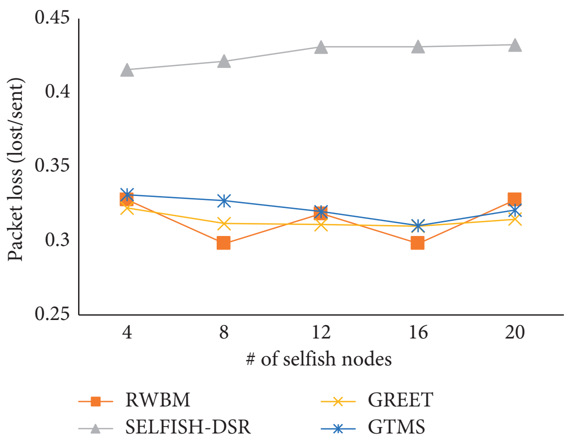


Figure 17: Packet loss ratio with an increased number of selfish nodes.

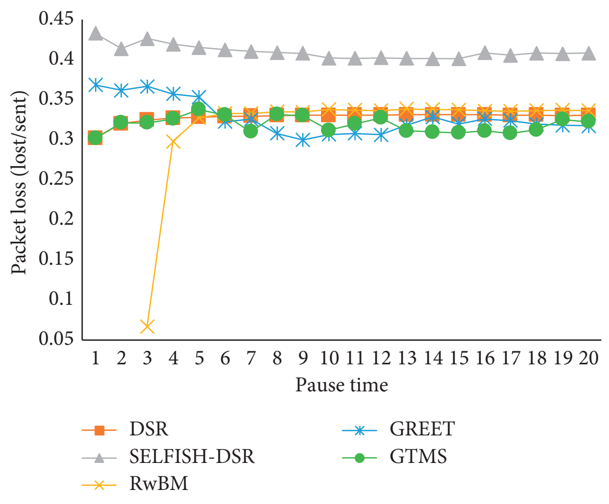


Figure 15: Packet loss ratio at twenty time pauses.

packet loss ratio for SELFISH-DSR is comparatively higher than other protocols. All other protocols are giving similar results in these experiments.

6. Conclusion

The existence of selfish nodes is widespread in wireless networks, particularly in ad hoc networks where an intelligent program or human controls the nodes. The nodes can be programmed to preserve their individual resources by not cooperating in the aggregate network goals. Several schemes have been introduced to overcome the issues of having such nodes in an ad hoc network. The selfish nodes can be managed either by blocking them or by stimulating them to participate in the network. In recent literature, credit-based incentive schemes are considered more effective and efficient for handling misbehaving or noncooperative nodes in ad hoc networks. Game theory is also used to realize the design and implementation of incentive-based schemes.

The incentive-based schemes can be used by opting for artificial neural networks (ANN) instead of the application of game theory. The repeated games can be replaced with an ANN module in which the neurons are trained with the passage of time. Moreover, the prevailing schemes can be interchangeably used in other forms of ad hoc networks. For example, a scheme designed for WSNs can be used in MANETs by handling the mobility or can be used in Internet of Things by considering various factors like heterogeneity and mobility. Moreover, the types of games can be changed in such schemes for better optimization of the network performance.

The simulation results show that the network performance is highly degraded without any mechanism for selfishness, as reflected by SELFISH-DSR protocol. The trust management scheme, GTMS, is giving acceptable results by addressing the target scenarios. However, it has relatively lower performance in almost all the experiments except the PDR and packet loss ratio experiments. GREET is outperforming in many experiments due to its advanced technique of letting the nodes understand the situation and adjust their cooperation level according to their benefits and the network needs. RwBM is also giving very good results due to its sophisticated technique by considering all the nodes' parameters and a state-of-the-art design of incentives for data communication. According to our experimental results, we can conclude that the selfishness of nodes can be managed by either the incentive-based or the evolutionary-based mechanisms. However, it must be noted that these experiments are made according to our understanding and cannot be considered as perfect. More work can be done in the analysis and comparisons of various protocols designed in the domain of selfish node management in ad hoc networks.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors acknowledge Prince Sultan University and Smart Systems Engineering Lab for their valuable support and provision of research facilities that were essential for completing this work. Furthermore, the authors acknowledge the support of PSU for paying the Article Processing Charges (APC) of this publication.

References

[1] N. Choudhury, R. Matam, M. Mukherjee, and L. Shu, "Beacon synchronization and duty-cycling in ieee 802.15.4 cluster-tree networks: a review," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1765–1788, 2018.

[2] D. G. Reina, M. Askalani, S. L. Toral, F. Barrero, E. Asimakopoulou, and N. Bessis, "A survey on multihop ad hoc networks for disaster response scenarios," *International Journal of Distributed Sensor Networks*, vol. 11, no. 10, Article ID 647037, 2015.

[3] M. M. Nasralla, N. Khan, and M. G. Martini, "Content-aware downlink scheduling for LTE wireless systems: a survey and performance comparison of key approaches," *Computer Communications*, vol. 130, pp. 78–100, 2018.

[4] C. Zhang and J. Wang, "Energy-efficient resource allocation for energy harvesting-based cognitive machine-to-machine communications," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 3, pp. 595–607, 2019.

[5] N. Choudhury, R. Matam, M. Mukherjee, and J. Lloret, "LBS: a beacon synchronization scheme with higher schedulability for ieee 802.15.4 cluster-tree-based IoT applications," *IEEE Internet of Things Journal*, vol. 6, no. 5, 2019.

[6] N. Choudhury and R. Matam, "Distributed beacon scheduling for IEEE 802.15. 4 cluster-tree topology," in *Proceedings of the 2016 IEEE Annual India Conference (INDICON)*, pp. 1–6, New York, NY, USA, December 2016.

[7] L. Buttyan and J.-P. Hubaux, "Enforcing service availability in mobile ad-hoc WANs," in *Proceedings of the First Annual Workshop on Mobile and Ad Hoc Networking and Computing. MobiHOC (Cat. No.00EX444)*, Catania, Italy, July 2000.

[8] A. Attiah, M. F. Amjad, M. Chatterjee, and C. Zou, "An evolutionary routing game for energy balance in Wireless Sensor Networks," *Computer Networks*, vol. 138, pp. 31–43, 2018.

[9] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[10] X. Wu, G. Chen, and S. K. Das, "Avoiding energy holes in wireless sensor networks with nonuniform node distribution," *IEEE Transactions On Parallel And Distributed Systems*, vol. 19, no. 5, pp. 710–720, 2008.

[11] G. S. Brar, S. Rani, V. Chopra, R. Malhotra, H. Song, and S. H. Ahmed, "Energy efficient direction-based PDORP routing protocol for WSN," *IEEE Access*, vol. 4, pp. 3182–3194, 2016.

[12] M. M. Nasralla, I. García-Magariño, and J. Lloret, "MASE-MUL: a simulation tool for MovementAware manet scheduling strategies for multimedia communications," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6651402, 12 pages, 2021.

[13] K. M. S. Huq, S. Mumtaz, J. Rodriguez et al., "Enhanced C-ran using D2D network," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 100–107, 2017.

[14] M. Mukherjee, R. Matam, L. Shu et al., "Security and privacy in fog computing: Challenges," *IEEE Access*, vol. 5, pp. 19293–19304, 2017.

[15] H. Gong, L. Yu, and X. Zhang, "Social contribution-based routing protocol for vehicular network with selfish nodes," *International Journal of Distributed Sensor Networks*, vol. 10, no. 4, Article ID 753024, 2014.

[16] S. N. Shah and R. H. Jhaveri, "A survey of various approaches to detect selfishness in wireless adhoc networks," in *Proceedings of the Green Computing and Internet of Things (ICGCIoT)*, Greater Noida, India, January 2015.

[17] J. Duan, D. Yang, H. Zhu, S. Zhang, and J. Zhao, "TSRF: a trust-aware secure routing framework in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 10, no. 1, Article ID 209436, 2014.

[18] M. M. Umar, S. Khan, R. Ahmad, and D. Singh, "Game theoretic reward based adaptive data communication in

wireless sensor networks," *IEEE Access*, vol. 6, no. 1, pp. 28073–28084, 2018.

[19] H.-Y. Shi, "Game theory for wireless sensor networks: a survey," *Sensors*, vol. 12, no. 7, Article ID 90559097, 2012.

[20] H. A. Muhammad, T. A. Yahiy, and N. Al-Salihi, "Comparative study between reactive and proactive protocols of (MANET) in terms of power consumption and quality of service," in *Proceedings of the International Conference on Computer Networks*, Madurai, India, December 2019.

[21] B. Chen, J.-L. Mao, N. Guo, G.-H. Qiao, and N. Dai, "An incentive detection mechanism for cooperation of nodes selfish behavior in wireless sensor networks," in *Proceedings of the 2013 25th Chinese Control and Decision Conference (CCDC)*, Guiyang, China, May 2013.

[22] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[23] K. Balakrishnan, J. Deng, and V. K. Varshney, "TWOACK: preventing selfishness in mobile ad hoc networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference*, New Orleans, LA, USA, May 2005.

[24] [. G. Kampitaki, E. D. Karapistoli, and A. A. Economides, "Evaluating selfishness impact on MANETs," in *Proceedings of the International Conference on Telecommunications and Multimedia*, Xi'an, China, July 2014.

[25] A. Mei and J. Stefa, "Give2Get: forwarding in social mobile wireless networks of selfish individuals," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 4, pp. 569–582, 2012.

[26] Z. Chen, Y. Qiu, J. Liu, and L. Xu, "Incentive mechanism for selfish nodes in wireless sensor networks based on evolutionary game," *Computers & Mathematics with Applications*, vol. 62, no. 9, pp. 3378–3388, 2011.

[27] M. Manjula and P. Elango, "A survey of selfish nodes behaviour in Mobile Adhoc network," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 4, no. 6, pp. 1848–1851, 2013.

[28] S. J. Borah, S. K. Dhurandher, I. Woungang, and V. Kumar, "A game theoretic context-based routing protocol for opportunistic networks in an IoT scenario," *Computer Networks*, vol. 129, pp. 572–584, 2017.

[29] J. F. Naves, I. M. Moraes, C. V. Albuquerque, and L. e. lrf, "Politicas de gerenciamento de buffer eficientes para redes tolerantes a atrasos e desconexoes," *Simp´osio Brasileiro de Redes de Computadores (SBRC 2012)*, vol. 15, pp. 293–305, 2012.

[30] C. Sinha, "Providing social likeness within a messaging context". U.S Patent 8600901, 3 December 2013..

[31] Q. Li, W. Gao, S. Zhu, and G. Cao, "A routing protocol for socially selfish delay tolerant networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1619–1632, 2012.

[32] Y. Li, G. Su, D. O. Wu, D. Jin, L. Su, and L. Zeng, "The impact of node selfishness on multicasting in delay tolerant networks," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 5, Article ID 22242238, 2011.

[33] J. M. Sánchez-Matamoros, J. M.-d. Dios, and A. Ollero, "Cooperative localization and tracking with a camerabased WSN," in *Proceedings of the 2009 IEEE International Conference on Mechatronics*, Málaga, Spain, April 2009.

[34] G. Brahim, A. Al-Fuqaha, M. Guizani, and B. Khan, "A model for cooperative mobility and budgeted QoS in MANETs with heterogenous autonomy requirements," in *Proceedings of the IEEE Global Telecommunications Conference, 2008. IEEE GLOBECOM*, New Orleans, Louisiana, December 2008.

[35] A. Al-Fuqaha, B. Khan, A. Rayes, M. Guizani, O. Awwad, and G. B. Brahim, "Opportunistic channel selection strategy for better QoS in cooperative networks with cognitive radio capabilities," *IEEE Journal On Selected Areas In Communications*, vol. 26, no. 1, pp. 156–167, 2008.

[36] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: privacy-preserving data aggregation in wireless sensor networks," in *Proceedings of the INFOCOM 2007. 26th IEEE International Conference on Computer Communications*, Washington, DC, May 2007.

[37] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the MOBICOM*, Article ID 255265, New York, NY, USA, 2000.

[38] R. A. Shaikh, H. Jameel, B. J. d'Auriol, H. Sungyoung Lee, and Y. J. Young-Jae Song, "Group-based trust management scheme for clustered wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 11, pp. 1698–1712, 2009.

[39] Q. He, D. Wu, and P. Khosla, "SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks," *IEEE Wireless Communications and Networking Conference*, vol. 2, 2004.

[40] K. A. Gemeda, G. Gianini, and M. Libsie, "An evolutionary cluster-game approach for wireless sensor networks in non-collaborative settings," *Pervasive and Mobile Computing*, vol. 42, 2017.

[41] B. Subba, S. Biswas, and S. Karmakar, "A game theory based multi layered intrusion detection framework for wireless sensor networks," *International Journal of Wireless Information Networks*, vol. 25, 2018.

[42] R. Raja and P. G. Kumar, "Designing a novel framework for evaluation of trust in mobile ad-hoc networks," *Journal of Computational and Theoretical Nanoscience*, vol. 15, no. 1, pp. 338–344, 2018.

[43] W. Li and H. Song, "ART: an attack-resistant trust management scheme for securing vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 960–969, 2015.

[44] S. B. Boudec and J.-Y. Le, "Performance analysis of the CONFIDANT protocol," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing - MobiHoc '02*, New York; NY; USA, July 2002.

[45] Y. B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent, "Trust management system design for the Internet of Things: a context-aware and multi-service approach," *Computers & Security*, vol. 39, pp. 351–365, 2013.

[46] H. Janzadeh, K. Fayazbakhsh, M. Dehghan, and M. S. Fallah, "A secure credit-based cooperation stimulating mechanism for MANETs using hash chains," *Future Generation Computer Systems*, vol. 25, no. 8, pp. 926–934, 2009.

[47] L. Yang, Y. Lu, L. Xiong, Y. Tao, and Y. Zhong, "A game theoretic approach for balancing energy consumption in clustered wireless sensor networks," *Sensors*, vol. 17, no. 11, p. 2654, 2017.

[48] Q. L. X. Yu, "An incentive mechanism game theory based for cooperation in wireless ad hoc networks," in *Proceedings of the 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Datong, China, October 2016.

[49] Q. Xu, Z. Su, and S. Guo, "A game theoretical incentive scheme for relay selection services in mobile social networks,"

IEEE Transactions on Vehicular Technology, vol. 65, no. 8, pp. 6692–6702, 2016.

[50] F. Bao, I.-R. Chen, M. Chang, and J.-H. Cho, "Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection," *IEEE Transactions On Network And Service Management*, vol. 9, no. 2, pp. 169–183, 2012.

[51] L. Buttyán and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Mobile Networks and Applications*, vol. 8, no. 5, pp. 579–592, 2003.

[52] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: a simple, cheat-proof, credit-based system for mobile adhoc networks," in *Proceedings of the IEEE INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, San Francisco, CA, USA, March 2003.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5523806, 11 pages
https://doi.org/10.1155/2021/5523806

Research Article

Poor Coding Leads to DoS Attack and Security Issues in Web Applications for Sensors

Khuda Bux Jalbani[1], **Muhammad Yousaf**[1], **Muhammad Shahzad Sarfraz**[2], **Rozita Jamili Oskouei**[3], **Akhtar Hussain**[4] **and Zojan Memon**[5]

[1]*Riphah Institute of Systems Engineering, Riphah International University, Islamabad 44000, Pakistan*
[2]*Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan*
[3]*Department of Computer Science and Information Technology, Islamic Azad University, Mahdishahr Branch, Mahdishahr, Iran*
[4]*Department of Information Technology, Quaid-E-Awam University of Engineering, Science and Technology, Nawabshah 67450, Pakistan*
[5]*Department of Information Technology, University of Sufism and Modern Sciences, Bhitshah 70140, Pakistan*

Correspondence should be addressed to Rozita Jamili Oskouei; rozita2020j@gmail.com

Received 21 January 2021; Revised 1 March 2021; Accepted 8 May 2021; Published 20 May 2021

Academic Editor: Shah Nazir

As the SQL injection attack is still at the top of the list at Open Web Application Security Project (OWASP) for more than one decade, this type of attack created too many types of issues for a web application, sensors, or any similar type of applications, such as leakage of user private data and organization intellectual property, or may cause Distributed Denial of Service (DDoS) attacks. This paper focused on the poor coding or invalidated input field which is a big cause of services unavailability for web applications. Secondly, it focused on the selection of program created issues for the WebSocket connections between sensors and the webserver. The number of users is growing to use web applications and mobile apps. These web applications or mobile apps are used for different purposes such as tracking vehicles, banking services, online stores for shopping, taxi booking, logistics, education, monitoring user activities, collecting data, or sending any instructions to sensors, and social websites. Web applications are easy to develop with less time and at a low cost. Due to that, business community or individual service provider's first choice is to have a website and mobile app. So everyone is trying to provide 24/7 services to its users without any downtime. But there are some critical issues of web application design and development. These problems are leading to too many security loopholes for web servers, web applications, and its user's privacy. Because of poor coding and validation of input fields, these web applications are vulnerable to SQL Injection and other security problems. Instead of using the latest third-party frameworks, language for website development, and version database server, another factor to disturb the services of a web server may be the socket programming for sensors at the production level. These sensors are installed in vehicles to track or use them for booking mobile apps.

1. Introduction

With the growing number of mobile app users, everyone is trying to develop their business apps as soon as possible. So these mobile apps are used to track the user's activities, getting information regarding vehicle locations, and tracking of logistics. For tracking vehicles, different types of mobile apps or sensors are used. For the full functionality of these apps or sensor devices, support software is required such as Personal Home Page (PHP) for the backend server, MySQL for data storage, and NodeJS for other required functionality as per requirements of clients or devices. Too many different types of open-source frameworks are used for backend functionality if the old version of these third-party tools is used. Then they may have a well-known vulnerability that can be exploited by an attacker or adversaries.

All of the above Structured Query Language (SQL) injection attacks are more dangerous for the web application or for other devices (like mobile apps or sensors) which are using it as web services. This attack is at the top of all injection-type family attacks or web application attacks. In this attack, the weakness of input fields is exploited by the attackers. It is performed by inserting the SQL query command into the input or the query will be appended with the targeted Uniform Resource Locator (URL). These SQL queries are transformed into SQL code which is inserted by an attacker [1, 2]. This injection vulnerability is the main point of web application security exploitation by an attacker. These loopholes in web application remain because the testing of input boxes is sanitized properly [1]. If the PHP old version is used during the development and testing phase, it will also make web applications vulnerable. A web application developed on a local system with the latest version of PHP and the old version installed on the production server may lead to the unavailability of web application services for users. This may disturb other applications during the upgradation of the PHP version if there is a shared server to save the operational cost of hosting or any old version of PHP framework such as WordPress in use that can be also more dangerous for web application services [3]. With these vulnerable frameworks, the attacker can delete databases or ask for a ransom to restore those databases or encrypted code. Another issue for the performance and security of web applications is the sockets used for communication between the web server and sensors. The nature of sensors is heterogeneous; due to this, the use of the same protocol for communication is not possible [4]. These sensors are more vulnerable to be exploited due to low computing and power storage. These sensors are used for the different types of services such as, in the health system for monitoring conditions of patients [5], in vehicles to trace the location, and in water management for checking the level of water at rivers, etc. As the usage of IoT sensors is growing in every area of life, the number of attacks also increase. These attacks are performed on different layers of IoT sensors to stop services for legitimate users, or to forward fake information. This research paper has used sensors in the vehicles for tracking them. The sensors are supported by socket-based communication for sending and receiving information from clients to the server. With the use of these sensors, it creates easiness for the monitoring of taxis and getting the information regarding peak hours more business areas for taxi services. As the number of requests increases and socket connections are going to backlog, then the server starts to stop binding port of sockets with Internet Protocol (IP) Address. To fix the error of port binding with IP address, this backlog should be cleared with two methods. First, in this paper, we have to kill all backlog connections manually or the second option is to restart the service of the socket connection program. As these two methods are performed, these services of sensor connections will be down for the users. It can be said it is a self-created Denial of Service DoS attack on services [6]. Due to this type of SQL injection attack, not only specific application is disturbed, but this complete database server is crashed so that all other applications are not accessible to valid users, which is the cause of DDoS attacks on databases servers. Another issue with sockets is that systems firewalls will be applied to make a socket with the server. It will be a more critical issue with the security of a web server because everyone is allowed to make a socket connection with the server and this can be exploited by an attacker for malicious activities.

This paper is described as follows: In Section 2, the related work is described for the most threatening web application attacks, sensor-based devices usage, and security issues, and WebSocket related problems. In Section 3, we described the proposed methodology. In Section 3.1, we will define how the database server is crashed with poor coding. In Section 4.2, we mentioned a few major issues related to WebSockets by using PHP programming. Section 4 will describe the results and discussions. In Section 5, this paper will be concluded.

2. Related Work

For one and half decade, the SQL injection attack was at top of all attacks on the web application. Every attacker targets the vulnerability of SQL injection in a web application to exploit them and take control of all services related to the webserver. This attack is performed by appending or inserting malicious SQL queries with a legitimate query. The author has proposed WAVES to test the vulnerabilities of SQL injection in web applications based on the black-box method [7]. This will find out every entry point of SQL injection in web applications by using the web crawling method. After that, it will use a predefined number of methods and attack techniques on those vulnerabilities for the SQL injection attack. In the last step, WAVES will monitor the web application traffic for checking the reaction of that attack, and for more betterment of the attack method, the machine learning method is used. The researcher has proposed the method for tautology-based SQL injection queries based on the application layer-created queries and should be analyzed with a static method by combining it with an automated process [8]. But it is limited to tautology-based SQL injection and will not detect or prevent attacks related to this. The AMNESIA model has been proposed by an author and it is based on static and dynamic analysis of the queries [9]. In the static analysis process, the AMNESIA looks into the application generated legitimate queries with every connection to the database; on this condition, it will create a prototype of queries. In the second process of dynamic analysis, AMNESIA interprets all queries; after that, these will be forwarded to the database and then it will be comparing every query with a static prototype model already created. If any query is out of scope from this model, then these queries will be considered as SQL injection attacks and these queries will not be executed on a database server. But this model has more ratio of false positive or false negative if queries are encrypted by developers. The ARDILLA tool has been proposed by an author for the detection of SQL injection and Cross-Site Scripting attacks in real-time [10]. The ARDILLA is developed for the PHP scripts input testing only and sessions are not handled by this tool. The Web Application SQL injection Protector (WASP) has been proposed by the researcher, and this tool is used to detect SQL injection attacks from stored procedures with real-time configuration [11]. But this tool needs much more improvement for the protection of web applications from SQL injection and XSS attacks.

As the demand for easy life increased due to this usage, IoT devices or sensors are also increased. Some people need to know about their business, such as tracking their goods, vehicles, Cab services and monitoring of patient's health conditions, etc. The latest version of Homecare is known as E-Homecare services which have functions of injection timings, diet management, a routine of exercise, and monitoring of health conditions [12]. "SmartPill" Wireless container is utilized to transmit intraluminal pH, pressure, and temperature information at standard interims to SmartPill GI Monitoring framework [13]. Titan implantable hemodynamic sensor (IHM) is a gadget having a size of a pencil eraser that can be embedded in the core of a patient to quantify basic factors like temperature, and afterward, remotely transmit this information to a protected database [14]. Intelligent vehicle: An arrangement of mechanical applications to gather data on the position, kinematics, and elements of the vehicle, the condition of nature, and the condition of the driver and the traveler, to survey such data and settle on choices dependent on it. It is fit for duplex correspondence with a side of the road foundation and different vehicles, to utilize computerized map applications and satellite situating frameworks, it has a functioning web association and its physical location [15]. A Smart Sustainable City (SSC) using Information and Communication Technologies (ICTs) for the creative city will give a better life, productivity of urban facilities, and competitiveness in between them. With this, current and future needs can be met as per economically, socially, and environmental changes [16]. The scholar in [17] proposed a 2-pivot MAG for distinguishing vehicle driving direction. A high discovery pace of 99% was seen when making a trip vehicles pass near the sensor. Execution corrupted to 89% as the signal-to-noise ratio (SNR) decreased. A two-edge, four-state machine calculation was presented in [18] for vehicle discovery utilizing 3-hub MAG.

The WebSocket protocol was created as a major aspect of the HTML 5 activity to encourage communications channels over TCP. WebSocket is neither a request/response nor a distribute/subscribe in the protocol. In WebSocket, a customer introduces a handshake with a server to set up a WebSocket session. The handshake itself is like HTTP, so web servers can deal with WebSocket sessions just as HTTP associations through a similar port [19]. As the WebSocket connection is established between client and server, they can send or receive data to each other with half-duplex. This connection will remain active with unlimited time and can be closed by the client or server as they want [20]. The Websocket Application Program Interface (API) provides great functionality to websites to establish a connection and transmit data to any server [21]. Due to this functionality, it is easy and effortless for a developer to work on WebSockets in websites for transmitting data. The major drawback of WebSocket that it does not add the HyperText Transfer Protocol (HTTP) header along with the connection. Due to this, the policy of origin resource verification does not provide a secure connection anymore because these origins can be spoofed [22]. As per the author, another security issue is cache poisoning with Websockets; to protect them from this vulnerability, the protocol working group introduced the method of frame-masking [21, 23]. With the addition of frame-masking, the cross-site scripting injection attack has been blocked, but the information of WebSocket cannot be transferred in plain text between client and server. The frame-masking has been used with WebSockets for the protection from cache poisoning attack, but it makes it harder the detection of malicious data via firewalls and other virus detection tools [24]. The firewalls can be bypassed by the attackers to compromise the targeted user browser and create that as a WebSocket proxy between him and the targeted organization network [25]. It is also vulnerable to another more common attack type of Denial of Service (DoS). In this case, attackers are trying to overwhelm the clients or server with bursts of information or maybe too many numbers of connections request; due to this, the legitimate users will not be able to complete their requests. In any case, on the web applications that use WebSockets, the XSS vulnerabilities open up a few new dangers. For example, with an XSS defencelessness, the attack might have the option to supersede the callback elements of a WebSocket association with custom ones [22]. This methodology permits the attacker to sniff the traffic, control the information, or actualize a man-in-the-middle attack against WebSocket associations.

When InnoDB is applied with MySQL creating too many issues related to SQL injection attacks which may lead to a complete crash of the database server. Therefore, there is a need for a solution for the prevention of this type of SQL injection attack.

3. Proposed Methodology

This research will describe the practical deployment of WebSockets for the tracking of a vehicle installed sensors in them. The complete deployment scenarios of the vehicle tracking application are defined in this section. To take care of major constraints regarding low power storage, these sensors have been implemented in idle state condition or can say it in passive mode sensors. This web application and MySQL server are deployed on Ubuntu 19.04 along with all updates of operating systems. And all other tools are also up-to-date as per deployment of this application, which is implemented about six months ago. Furthermore, in this research proposed method the latest PHP version 7.3, Laravel framework 5.4.36, MySQL 5.7, and NodeJS v13.3.0 are installed for the proposed application (see Figure 1).

The aim is to avoid the well-known vulnerabilities regarding the operating system, web application framework, database server, and version of PHP used for WebSockets. To avoid fake sensors, the authentication process has been implemented with the help of the Laravel framework web application. In this process, the drivers or vendors of vehicles need to be registered at web application along with their personal information and sensor identification number, which may be its serial number. As the constraint of sensors, these are accepting WebSockets only for communication

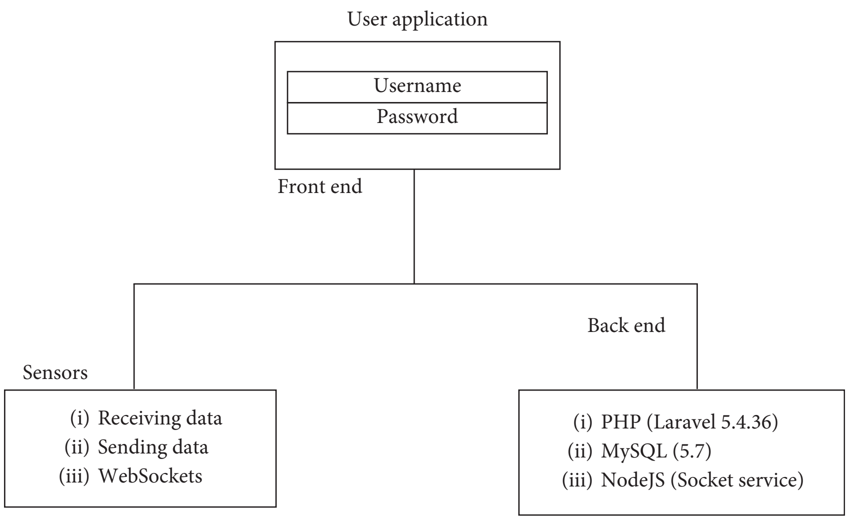
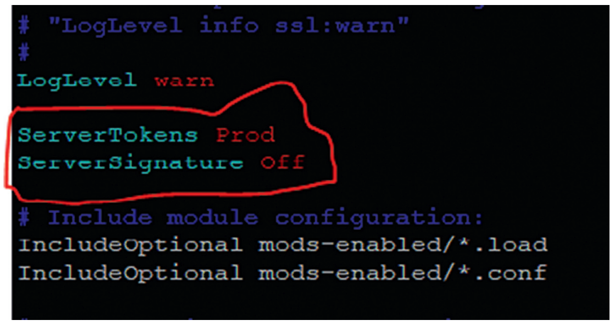


Figure 1: Vehicle tracking system overview.

instead of any API. So for this communication between vehicles and servers, the WebSocket program is developed in custom PHP, which is defined in the results and discussions section. For the security of web applications at the operating system level, iptables or Uncomplicated Firewall (UFW) has been used to block unauthorized users. For more protection at the system level, the version of the apache web server and operating system is configured as hidden in apache2.conf (see Figure 2) with red circle options.

The user's login information, sensor details, and movement of vehicles are stored in the MySQL database, which is also hosted on the same server along with a web application. For the optimization of the MySQL database, in this research, the InnoDB has been used for the stored procedure, which gives the functionality of foreign key relationships between tables. The more features and drawbacks of InnoDB and MyISAM stored procedure are explained in the next section of the crashing database server. All critical information regarding users, sensors, and vehicles is stored in a database, so the implemented security of it at the system level such as disallow remote login on a database for the root or normal users from any IP address. The default databases in MySQL have been removed, and the database users have been created with complex password authentication to avoid the brute-force attacks on MySQL databases. And for the protection from cache poisoning or man-in-the-middle attack, encryption has been implemented for the WebSocket communication between sensors and with a web application server. To compare the WebSocket issues regarding performance and backlog closing of connections with sensors, the NodeJS has been implemented for it.



```
# "LogLevel info ssl:warn"
#
LogLevel warn

ServerTokens Prod
ServerSignature Off

# Include module configuration:
IncludeOptional mods-enabled/*.load
IncludeOptional mods-enabled/*.conf
```

Figure 2: Hiding the operating system and Apache version.

3.1. Crashing Database Server. The most critical part of any web application is its databases because it is the main source of information storage regarding its users, user sessions, integrated third-party applications information, financial information, locations of users or vehicles tracking information, and much more. As per the last two-decade research work regarding web application attacks and OWASP top 10 attack reports of 2013 and 2016 [26, 27], the SQL injection attack is at top of all. This attack is more dangerous for web applications in the form of information stealing, DoS attack [28], system crashing, alteration in database records to insert the fake information, traffic redirection, and getting root rights of system. This attack is easy to be performed by attackers with little effort. That is why everyone is trying to exploit this injection vulnerability. The SQL injection attack is performed on web applications that have the vulnerability of weak validation on input fields such as login form. These input fields are not sanitized properly. For the better performance and optimization of MySQL database, two types of stored procedures are used, namely InnoDB and MyISAM. These methods' usage is based on the requirements of web applications. The advantages and disadvantages are explained as follows.

3.2. InnoDB Store Procedure. For the transactional database or relationships of tables, the InnoDB stored procedure is used [29]. This is used for more write operations into databases such as insert and update. This stored procedure is used for solving the issue of table-locking weakness. The InnoDB is used in applications where data integrity is in high demand for the users, and this is achieved with the help of relationship and transaction functionality. It is used for faster write operations into databases because it supports locking the tables at row-level for better integrity. It is the most fitting stored procedure for high-simultaneousness and high-exchange remaining tasks at hand.

3.3. MyISAM Store Procedure. The default stored procedure for MySQL is MyISAM used for the high usage of reading operations. But another issue with this is that the less transactional and low level of concurrent write operations are supported. If any application needs big-size tables and fewer changes are required, then MyISAM stored procedure is used as a priority [29]. If anyone wants to use it as transactional, then they need to add an extra MySQL SQL extension of Lock Table and Unlock Tables. It is used for the high speed read and simple in implementation due to this most popular for general-purpose stored usage.

In this research paper, we have used the InnoDB stored procedure for the vehicle tracking application. In this application, the relationships between tables and more write operations are needed. The user login information, sensor information, vendors' details, and the location tracking of vehicles are stored in this database. As per the previous study of the SQL injection attack, sanitized input fields for malicious query protection used the latest version of framework and MySQL. But still, the database server has been crashed with one wrong value entry at the user login page. That wrong password value with special characters is in bold (see Figure 3).

In 2008 or early for bypassing HTTP communication, a new method for two-way communication has been developed. Maltreatment of HTTP for bidirectional correspondence prompts imperfect utilization of HTTP connections, causing superfluous issues for correspondence parties. For the solution of this issue, it has been added into working draft 10 for the HTML5 in June 2008 and that program function was named TCP connection which is based on Transmission Control Protocol (TCP) socket API [30]. The TCP connection was renamed WebSocket in late July 2008. Originally the WebSocket was created by the World Wide Web Consortium (W3C) and the WHATWG group, but it was transferred to Internet Engineering Task Force (IETF) for further development in February 2010. As the too many numbers revisions, IEFT published the final version as a WebSocket protocol with Request For Comments (RFC) 6455 in December 2011 [31]. The communication methods are used [32] given below.

3.4. Request or Response Method. It is a system where the customer sends a solicitation to the server and gets a reaction. This procedure is driven by some cooperation, for example, the snap of a catch on the website page to invigorate the entire page. At the point when Asynchronous JavaScript and XML (AJAX) entered the image, it made the website pages' dynamic through the use of JavaScript mechanization and aided in stacking some piece of the page without stacking the entire page once more. When InnoDB is applied with MySQL creating too many issues related to SQL injection attacks, it may lead to a complete crash of the database server. Therefore, there is a need for a solution for the prevention of this type of SQL injection attack.

```
{"id":3,"name":"mobile","email":"mobile@admin.com","password":"70U79Ee\/P
FHu2","role_id":2,"remember_token":null,"status":1,"updated_at":"2020-01-27
08:02:43","created_at":"2019-11-04 00:00:00"}
```

Figure 3: Wrong value entered in password field.

3.5. Polling Method. It is a system for situations where information should be invigorated without client collaboration, for example, the score of a football coordinate. In surveying, the information is brought after a set timeframe and it continues hitting the server, whether or not the information has changed or not. This makes superfluous solicitations the server, opening an association and afterward shutting it without fail. It is related to WebSockets that shows how they handle the request of users.

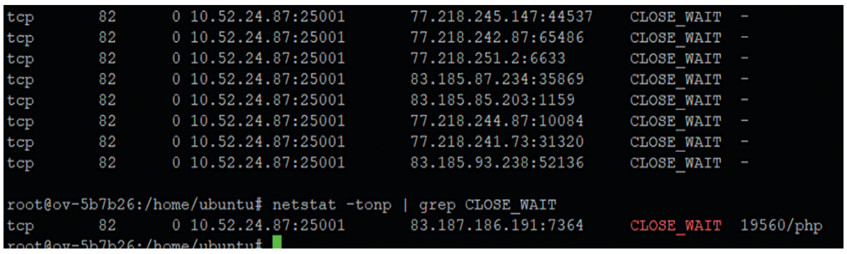
3.6. Long Polling Method. It is an instrument mishandling Request/Response where the association is kept open for a specific timeframe. At the point when the customer utilizes long surveying, the server reacts to the customer simply after the information is fit to be sent, which contrasts with the conventional Request/Response strategy where the reaction is sent to the customer directly after the solicitation. This is one of the approaches to accomplish constant correspondence. However, it works just with known time interims.

This research has used the PHP custom program for the WebSocket communication between vehicle sensors and web application server. This connection is used for the tracking of vehicles to get more information regarding peak hours of passengers for taxis and movement of vehicle information for their vendors. The connection between the web server and vehicle sensors has no time limit to close that connection as the few logs of the CLOSE_WAIT state are given (see Figure 4).

As in the above log entries regarding CLOSE_WAIT state or it is known as long-polling of WebSocket connection are given there are too many more connections in this state. It is causing too many problems for the webserver. The main issue of this the IP address cannot be bind with port 25001 of WebSocket for new connection requests for sensors. And sending or receiving of information from vehicle sensors is also stopped due to this issue of IP address binding. The Websocket connection creation code and temporary solution to this custom PHP program and permanent solution of this problem are discussed in the next section of results and discussion.

4. Results and Discussion

This section will discuss the issue of wrong entry from the user into SQL databases which crashed its InnoDB store procedure, how the WebSocket connections are created in the PHP custom program, and the issue of CLOSE_WAIT state of those connections for an unlimited time. The temporary solution to this problem is to apply the timer for unused opened connections to closing those WebSocket



```
tcp        82      0 10.52.24.87:25001       77.218.245.147:44537    CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       77.218.242.87:65486     CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       77.218.251.2:6633       CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       83.185.87.234:35869     CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       83.185.85.203:1159      CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       77.218.244.87:10084     CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       77.218.241.73:31320     CLOSE_WAIT  -
tcp        82      0 10.52.24.87:25001       83.185.93.238:52136     CLOSE_WAIT  -

root@ov-5b7b26:/home/ubuntu# netstat -tonp | grep CLOSE_WAIT
tcp        82      0 10.52.24.87:25001       83.187.186.191:7364     CLOSE_WAIT  19560/php
root@ov-5b7b26:/home/ubuntu#
```

Figure 4: WebSocket CLOSE_WAIT state.

connections. A permanent solution to this problem is the usage of NodeJS-based application for the unlimited WebSocket connections without any overhead on the server. As per best knowledge, this research has used the latest software and tools for this web application of the vehicle's tracking system to avoid known vulnerabilities of SQL injection, PHP frameworks, Apache web server, and any other vulnerability related to the operating system. First, we will discuss how the MySQL database server has been crashed with a single entry at the login page.

4.1. InnoDB Crashed. The InnoDB store procedure is used for the transactional operations and relationships between tables. It is used in web applications on which write operations are performed more frequently with the support of table lock for the integrity of data. But database server has been crashed with a single wrong entry into the database at the login page as described in Figure 3. Due to that wrong entry, the InnoDB store procedure has been crashed. The MySQL InnoDB crash is shown in Figure 5.

As in Figure 5, it is crashed due to the wrong value that has been inserted into databases. It was the main reason behind the crash of it. The value of key_buffer_size is inserted in large size from its normal value. Because of this reason, the InnoDB has been crashed. Figure 6 is given for more details regarding the database crash.

To recover from this issue, we have applied two methods: First, this research proposed changing the store procedure from InnoDB to MyISAM, and secondly, changing the value of my.cnf file to recovery mode for InnoDB. As the database structure changed from InnoDB to MyISAM, all relationships between tables have been deleted with this operation and transactional operations for insertion are also disturbed. This process has taken down web service for the sensors to track the location of vehicles and it is a shared hosting server. Due to this, other web application are also unavailable for the users. This is known as a self-DoS attack on organization's web services to clients. As in Figure 7, the recovery mode entry in my.cnf file has been added for a temporary solution to crashed InnoDB.

The InnoDB has been set as in recovery mode to fix the crashing issue of the store procedure. But due to these changes, the insertion, updating, and deletion operations have been locked into databases. All databases with the InnoDB store procedure have been disturbed due to these changes. The users of those web applications are unable to write data into a database. To recover from this issue, the MySQL database server has been reinstalled after getting a backup of all databases on this server. As experienced, the single wrong value has been entered by a user at the login page that has created this problem. To protect the database from this issue again, we have applied a limit of value on that field of the login page. And go through again for validation of each input filed of web application against any malicious record entry.

4.2. WebSocket Long Polling Issue. As in this research, as earlier mentioned in section V regarding states of WebSockets for communication between client and server. The first two states of request/response and poling are working fine in the PHP custom program for connection between vehicle sensors and web application for tracking. The code section for WebSocket connection creation between sensors and web applications (see Figure 8).

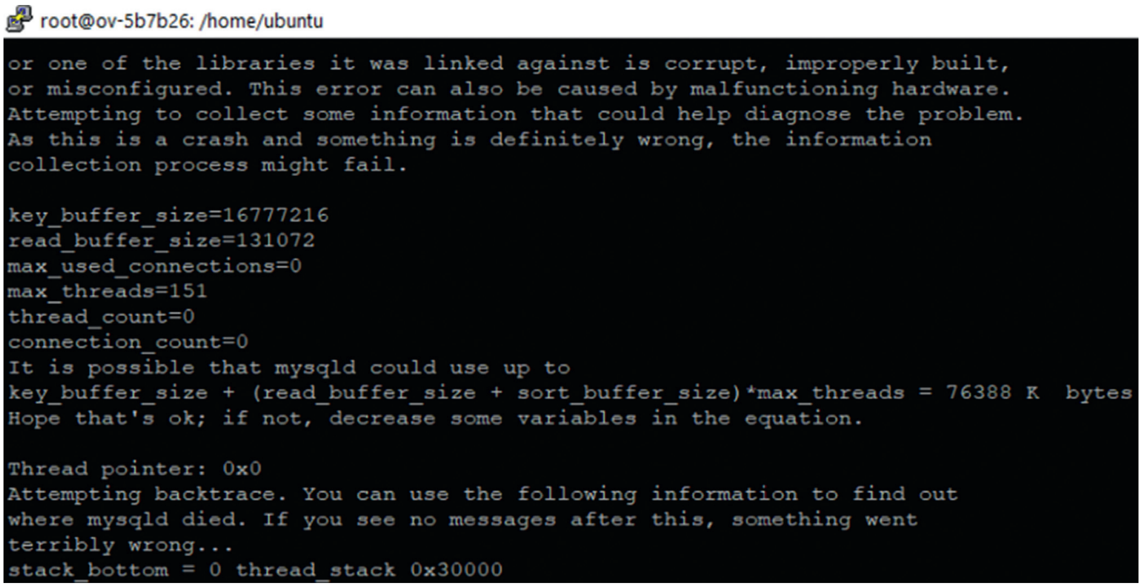
If a WebSocket connection is already created with the binding of the same port with IP address, then it shows us the message of WebSocket cannot be created. The connection creation time has been set to 300 seconds in decremented order. As the connection is successful, the host IP address and port will be a bind. Communication will be started between sensors and web servers for the storage of information regarding the tracking of vehicles or insurance details, etc.

We have faced issues at the long-polling connection. They are going into the backlog for an unlimited period. As in this research shown in Figure 3 in Section 6, there are too many connections in the state of CLOSE_WAIT and due to this, the new connection cannot be created and old connections are unable to send or receive required data. This problem occurs as more than 10 users are trying to connect with the webserver at the same time. This is not good for the production servers as in real-time, there are 0.3 million users who will have to use this service. For sharing their location information, insurance details, and other required details for the security of users. The issue of IP address binding with port occurred as the number of users is increasing as in Figure 9. We have just given a single error message of IP address bind, but there are too many numbers of the same type of error messages regarding binding.

To avoid this, IP address binding with the port of WebSocket has been applied a temporary solution. In this research, we have created a service for WebSocket connections (see Figure 10).

This service has been added into crontab job scheduling and this service has been restarted after every two hours to kill the backlog of WebSocket connection (see Figure 11).

By doing this, the service of WebSocket connection to users is unavailable because there is a need to kill all the

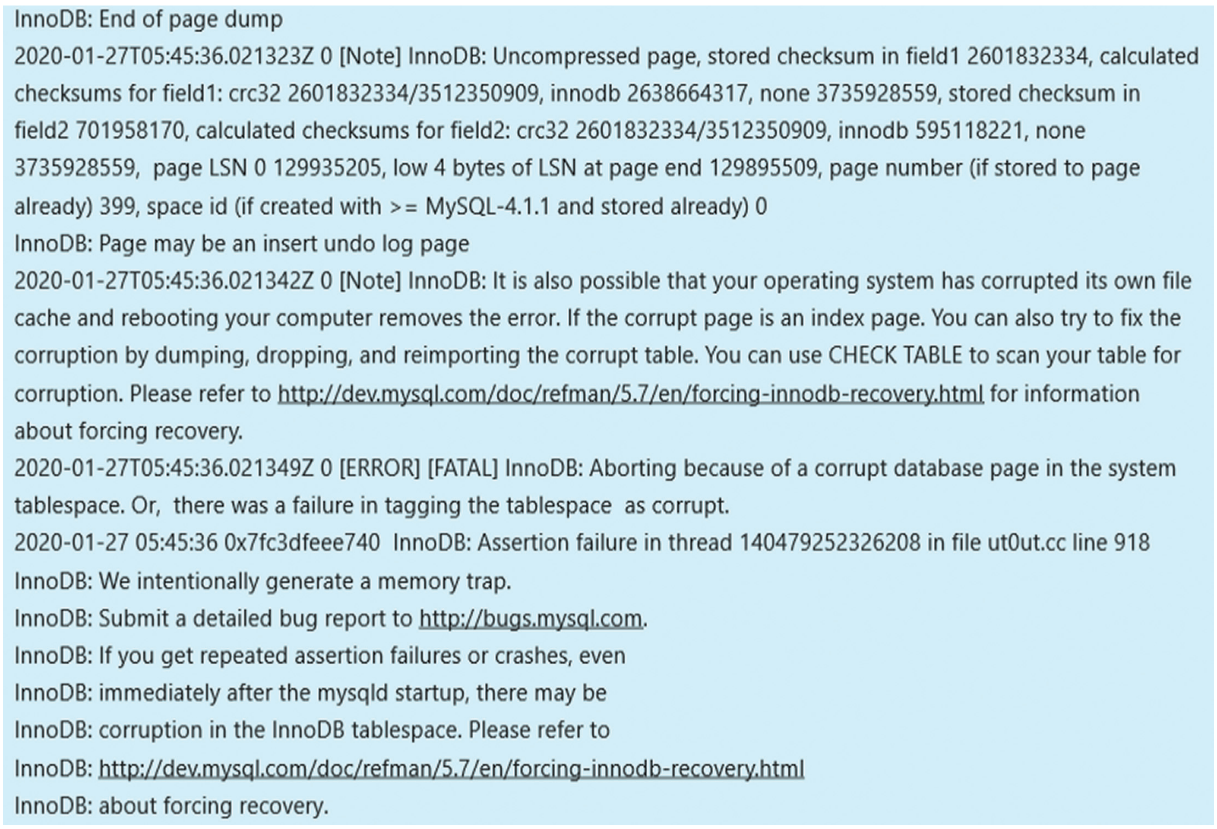


```
root@ov-5b7b26: /home/ubuntu
or one of the libraries it was linked against is corrupt, improperly built,
or misconfigured. This error can also be caused by malfunctioning hardware.
Attempting to collect some information that could help diagnose the problem.
As this is a crash and something is definitely wrong, the information
collection process might fail.

key_buffer_size=16777216
read_buffer_size=131072
max_used_connections=0
max_threads=151
thread_count=0
connection_count=0
It is possible that mysqld could use up to
key_buffer_size + (read_buffer_size + sort_buffer_size)*max_threads = 76388 K  bytes
Hope that's ok; if not, decrease some variables in the equation.

Thread pointer: 0x0
Attempting backtrace. You can use the following information to find out
where mysqld died. If you see no messages after this, something went
terribly wrong...
stack_bottom = 0 thread_stack 0x30000
```

Figure 5: InnoDB crashed with wrong entry.

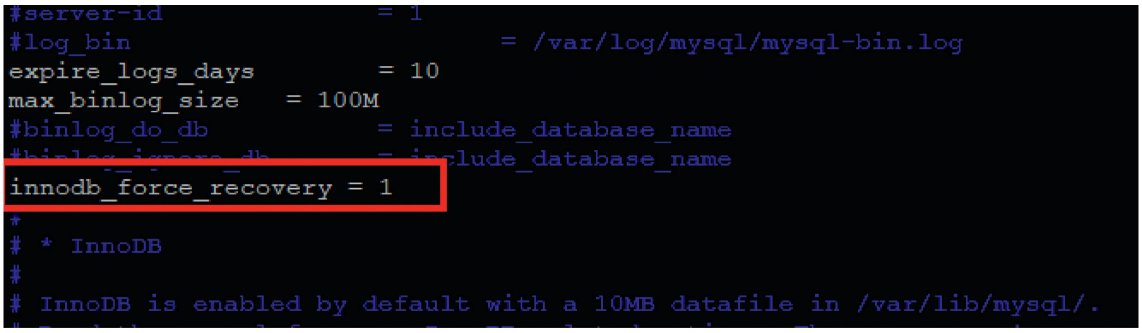


```
InnoDB: End of page dump
2020-01-27T05:45:36.021323Z 0 [Note] InnoDB: Uncompressed page, stored checksum in field1 2601832334, calculated
checksums for field1: crc32 2601832334/3512350909, innodb 2638664317, none 3735928559, stored checksum in
field2 701958170, calculated checksums for field2: crc32 2601832334/3512350909, innodb 595118221, none
3735928559,  page LSN 0 129935205, low 4 bytes of LSN at page end 129895509, page number (if stored to page
already) 399, space id (if created with >= MySQL-4.1.1 and stored already) 0
InnoDB: Page may be an insert undo log page
2020-01-27T05:45:36.021342Z 0 [Note] InnoDB: It is also possible that your operating system has corrupted its own file
cache and rebooting your computer removes the error. If the corrupt page is an index page. You can also try to fix the
corruption by dumping, dropping, and reimporting the corrupt table. You can use CHECK TABLE to scan your table for
corruption. Please refer to http://dev.mysql.com/doc/refman/5.7/en/forcing-innodb-recovery.html for information
about forcing recovery.
2020-01-27T05:45:36.021349Z 0 [ERROR] [FATAL] InnoDB: Aborting because of a corrupt database page in the system
tablespace. Or,  there was a failure in tagging the tablespace  as corrupt.
2020-01-27 05:45:36 0x7fc3dfeee740  InnoDB: Assertion failure in thread 140479252326208 in file ut0ut.cc line 918
InnoDB: We intentionally generate a memory trap.
InnoDB: Submit a detailed bug report to http://bugs.mysql.com.
InnoDB: If you get repeated assertion failures or crashes, even
InnoDB: immediately after the mysqld startup, there may be
InnoDB: corruption in the InnoDB tablespace. Please refer to
InnoDB: http://dev.mysql.com/doc/refman/5.7/en/forcing-innodb-recovery.html
InnoDB: about forcing recovery.
```

Figure 6: InnoDB crashed detailed logs.

processes related to these connections for sensors. This is not good for production servers or applications that the service of Websocket will be restarted after every few hours and critical for the real-time application, it is not considered as good practice in the case of the vehicle tracking system.

4.3. Permanent Solution for WebSocket Connections. To solve this issue, we have implemented WebSocket connections between sensors and webserver by using NodeJS 13.3.0 version. This research has faced the issue of the backlog in the state CLOSE_WAIT for too many number connections



```
#server-id              = 1
#log_bin                        = /var/log/mysql/mysql-bin.log
expire_logs_days        = 10
max_binlog_size   = 100M
#binlog_do_db           = include_database_name
#binlog_ignore_db       = include_database_name
innodb_force_recovery = 1
#
# * InnoDB
#
# InnoDB is enabled by default with a 10MB datafile in /var/lib/mysql/.
```

Figure 7: InnoDB recovery mode settings.

```
//$socket = socket_create (AF_INET, SOCK_STREAM, SOL_TCP) or die ("Could not
create socket\n");
        if (! ($socket = socket_create (AF_INET, SOCK_STREAM, 0))) {
            $errorcode = socket_last_error ();
            $errormsg = socket_strerror ($errorcode);

            die ("Couldn't create socket: [$errorcode] $errormsg");v
        }
        $timeout = array ('sec'=>3000, 'usec'=>0);
        $try = socket_set_option ($socket, SOL_SOCKET, SO_RCVTIMEO, $timeout);
        // bind socket to port
        $result = socket_bind ($socket, $host, $port) or die ("Could not bind to socket\n");
        // start listening for connections
        //$result = socket_listen ($socket, SOMAXCONN) or die ("Could not set up socket listener\n");
        $result = socket_listen ($socket, 4096) or die ("Could not set up socket listener\n");
```

Figure 8: Code in PHP for WebSocket connections.

```
[Mon Dec 16 14:34:09.096356 2019] [php7:warn] [pid 31393] [client
58.65.132.206:16043] PHP Warning: socket_bind(): unable to bind address [98]:
Address already in use in /home/benin/public_html/serverUpEncyption.php
```

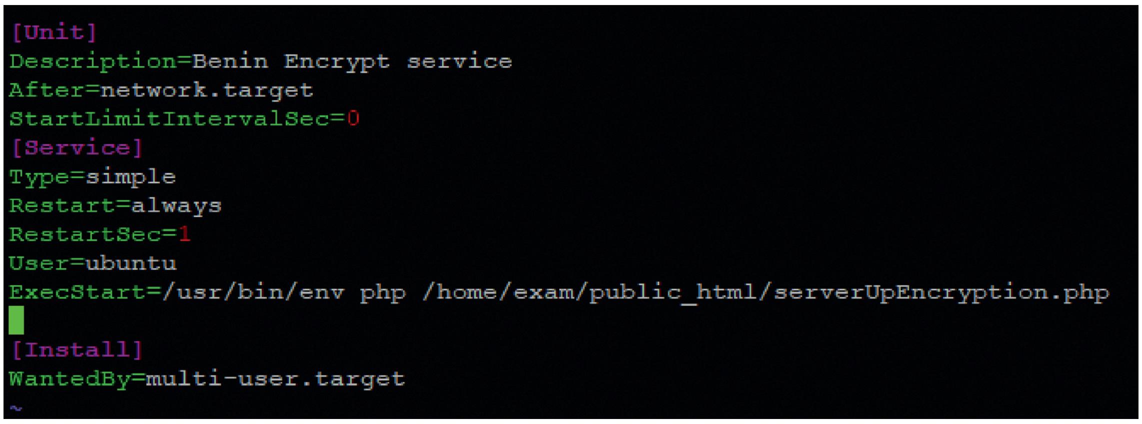
Figure 9: WebSocket binding error.

due to which new connection request is not completed. And old connections are unable to send or receive data for tracking the vehicles. The connection code in NodeJS is as shown in Figure 12.

In the above connection, the function has been created for the WebSocket for the sensors installed in vehicles. With the help of NodeJS, the autotest of more than 100K requests for the WebSocket connection has been created without any issue of backlog or error of IP address binding with the port (see Figure 13).

And in production currently, almost 5K requests are handled by the server without any error of binding port with IP address. The long-polling connections are not opened for the unlimited time between sensors and webserver. As the data regarding vehicle location tracking, its insurance details, and vendor details are shared or inserted into the database, the connection will be closed.

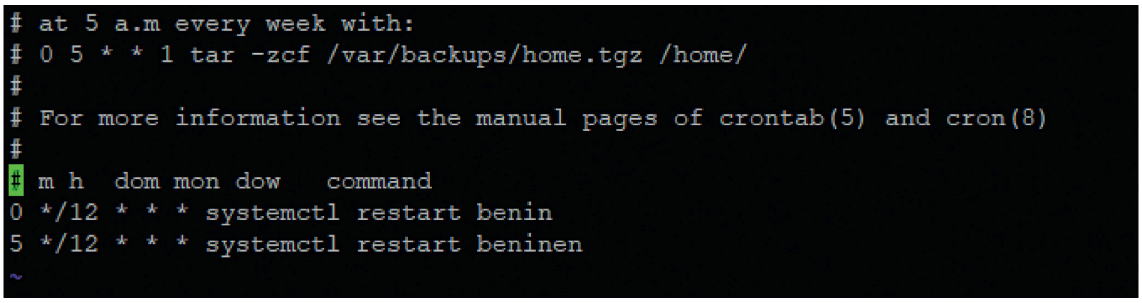
The confusion matrix has been given in Table 1 for the proposed methodology in this research paper. The second name of the confusion matrix is the error matrix which is used for the quantity-based analysis of static data. The proposed system to change from the structure of the MySQL database from InnoDB to MyISAM is best against the attack mentioned in the above sections. The overall accuracy of this proposed method is 96.154%.



```
[Unit]
Description=Benin Encrypt service
After=network.target
StartLimitIntervalSec=0
[Service]
Type=simple
Restart=always
RestartSec=1
User=ubuntu
ExecStart=/usr/bin/env php /home/exam/public_html/serverUpEncryption.php

[Install]
WantedBy=multi-user.target
~
```

Figure 10: WebSocket connection service.

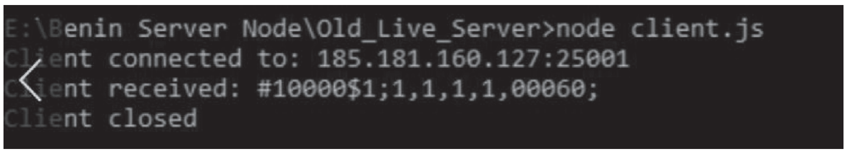


```
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
0 */12 * * * systemctl restart benin
5 */12 * * * systemctl restart beninen
~
```

Figure 11: WebSocket connection service restart.

```
var server = net.createServer (function (connection) {
  console.log ('client connected');
  connection.on ('data', function (data) { /* Logic Here */
          var det = data.toString (); // Decrypting Data
          det = decrypt (det);
          var sql = "INSERT INTO checkrequest (entriessent, timestamp) VALUES (?, ?)";
          var todayDate = getTodayDate ();
          con.query (sql, [det, todayDate]);
        console.log ('client data received: '+data);
          console.log ("Decrypted by Server: "+det);
          if (det == null){
                var responseToBeSent = "#####$1;1, 1, 1, 1, 86400;";
                responseToBeSent = encrypt (responseToBeSent);
                console.log ("Response Sent To Client: "+responseToBeSent);
                connection.write (responseToBeSent);
                console.log ("Response Sent");
      connection.destroy ();
          }
          else { /* data explode */
            var decodedVal = det.split (", ");
                if (decodedVal.length < 8 || decodedVal.length>8){
                    var responseToBeSent = "#####$1;1, 1, 1, 1, 86400;";
                    responseToBeSent = encrypt (responseToBeSent);
                    console.log ("Response Sent To Client: "+responseToBeSent);
                    connection.write (responseToBeSent);
                    console.log ("Response Sent");
                    connection.destroy ();
                }
}
```

Figure 12: WebSocket connection in NodeJS.



```
E:\Benin Server Node\Old_Live_Server>node client.js
Client connected to: 185.181.160.127:25001
Client received: #10000$1;1,1,1,1,00060;
Client closed
```

Figure 13: Autotest of WebSocket connections.

Table 1: Confusion matrix for the proposed method.

	Normal traffic	SQL injection	DDoS attacks	Classification overall	Precision
Normal traffic	600	0	0	600	100%
SQL injection	0	800	30	830	96.386%
DDoS attacks	0	50	600	650	92.308%
Truth overall	600	850	630	2080	
User accuracy	100%	94.118%	95.238%		

Overall accuracy = 96.154%.

5. Conclusions

For ease of business and to facilitate the customer's everyone wants his existence on web applications and mobile apps. As the trend of monitoring increased for security reasons and to get more data for traffic jams, peak hours for customers are hiring taxis, delivery services, health services, or online education, etc. Due to this, the usage of IoT devices is also increased. With the use of these devices, some existing security and some new issues have been arising such as for communication, WebSockets has been introduced in back 2008 and existing type of attacks is SQL injection. As have experienced just the latest tools, frameworks or operating system is not a solution to security breaches for a web application or sensors devices. But there is another factor with service unavailability, which is poor coding and selection of poor programming software solution of WebSocket connections between sensors and the webserver. As the high-security risk to the web applications is an SQL injection attack, it is performed on web applications that are not sanitized properly for input fields as we have seen that just a single wrong value entered at the login page crashed the MySQL InnoDB store procedure. Due to this, the services vehicles tracking to clients remain unavailable for a long time. To come online again temporarily, the stored procedure has been changed from InnoDB to MyISAM. But with this change, the performance of transactional operation decreased and relationships between tables also deleted. There is a need to change the mode MySQL for InnoDB into recovery mode. Due to this, other hosted websites also go down. For the protection from the injection type of attacks on web applications, the input fields need to be sanitized so that malicious users should be unable to insert malicious scripts into targeted web applications. Secondly, the WebSocket connection program was written in the PHP custom program, which has created another issue of the binding IP address with ports. The new connections of WebSocket are not created and old connections are also unable to send or receive data. For a temporary solution, we have implemented WebSocket connection service restart in CronJob of Ubuntu. And this was not a good solution for the production server. So, this research has changed the program for WebSocket connections which is NodeJS as a permanent solution to this issue. Now webserver can handle 100K+ requests without any problem of IP address binding with port numbers.

Data Availability

Data used to support this study are available from the corresponding author upon request via email (bux.khuda@gmail.com).

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] W. G. Halfond, J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," in *Proceedings of the IEEE International Symposium on Secure Software Engineering*, vol. 1, pp. 13–15, Washington, DC, USA, March 2006.

[2] A. Tajpour, M. Z. Heydari, M. Masrom, and S. Ibrahim, "SQL injection detection and prevention tools assessment," in *Proceedings of the 2010 3rd ICCOINS International Conference*

on Computer Science and Information Technology IEEE, vol. 9, pp. 518–522, Chengdu, China, July 2010.

[3] J. Ruohonen, "A demand-side viewpoint to software vulnerabilities in WordPress plugins," in *Proceedings of the Evaluation and Assessment on Software Engineering*, pp. 222–228, Copenhagen, Denmark, April 2019.

[4] C. Tian, X. Chen, D. Guo, J. Sun, L. Liu, and J. Hong, "Analysis and design of security in the internet of things," in *Proceedings of the 8th International Conference on Biomedical Engineering and Informatics (BMEI)*, pp. 678–684, IEEE, Shenyang, China, October 2015.

[5] M. Talal, A. A. Zaidan, B. B. Zaidan et al., "Smart home-based IoT for real-time and secure remote health monitoring of triage and priority system using body sensors: multi-driven systematic review," *Journal of Medical Systems*, vol. 43, no. 3, p. 42, 2019.

[6] D. Surbhi and K. Deepak, "Analysis of tree-based classifiers for web attack detection," in *Advances in Signal and Data Processing*, pp. 421–428, Springer, Singapore, Asia, 2021.

[7] Y. W. Huang, S. K. Huang, T. P. Lin, and C. H. Tsai, "Web application security assessment by fault injection and behavior monitoring," in *Proceedings of the 12th International Conference on World Wide Web*, pp. 148–159, Budapest, Hungary, May 2003.

[8] G. Wassermann and Z. Su, "An analysis framework for security in web applications," in *Proceedings of the FSE Workshop on Specification and Verification of Component-Based Systems (SAVCBS)*, pp. 70–78, Newport Beach, CA, USA, October 2004.

[9] W. G. Halfond and A. Orso, "AMNESIA: analysis and monitoring for neutralizing SQL-injection attacks," in *Proceedings of the 20th International Conference on Automated Software Engineering IEEE/ACM*, pp. 174–183, Long Beach, CA, USA, November 2005.

[10] A. Kieyzun, P. J. Guo, K. Jayaraman, and M. D. Ernst, "Automatic creation of SQL injection and cross-site scripting attacks," in *Proceedings of the 31st International Conference on Software Engineering IEEE*, pp. 199–209, Vancouver, Canada, May 2009.

[11] N. S. Ali and A. S. Shibghatullah, "Protection web applications using the real-time technique to detect structured query language injection attacks," *International Journal of Computer Applications*, vol. 149, no. 6, pp. 26–32, 2016.

[12] G. Demiris and J. Tan, *Rejuvenating Home Health Care and The-Home Care. E-Health Care Information Systems: an Introduction for Students and Professionals*, Jossey-Bass, San Francisco, CA, USA, 2005.

[13] S. Maqbool, H. P. Parkman, and F. K. Friedenberg, "Wireless capsule motility: comparison of the smartPill GI monitoring system with scintigraphy for measuring whole gut transit," *Digestive Diseases and Sciences*, vol. 54, no. 10, pp. 2167–2174, 2009.

[14] R. L. Benza, A. Raina, W. T. Abraham et al., "Pulmonary hypertension related to left heart disease: insight from a wireless implantable hemodynamic monitor," *The Journal of Heart and Lung Transplantation*, vol. 34, no. 3, pp. 329–337, 2015.

[15] F. Duchoň, P. Hubinský, J. Hanzel, A. Babinec, and M. Tölgyessy, "Intelligent vehicles as robotic applications," *Procedia Engineering*, vol. 48, pp. 105–114, 2012.

[16] S. N. Kondepudi, V. Ramanarayanan, A. Jain et al., *Smart Sustainable Cities: an Analysis of Definitions; the ITU-T Focus Group for Smart Sustainable Cities*, International Telecommunication Union (ITU), Geneva, Switzerland, 2012.

[17] N. Wahlström, R. Hostettler, F. Gustafsson, and W. Birk, "Classification of driving direction in traffic surveillance using magnetometers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1405–1418, 2014.

[18] K. Liu, H. Xiong, and H. He, "A new method for detecting traffic information based on anisotropic magnetoresistive technology," in *Proceedings of the Fifth International Conference on Machine Vision (ICMV 2012)*, Wuhan, China, March 2013.

[19] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, and J. Alonso-Zarate, "A survey on application layer protocols for the internet of things," *Transaction on IoT and Cloud Computing*, vol. 3, no. 1, pp. 11–17, 2015.

[20] A. Wessels, M. Purvis, J. Jackson, and S. Rahman, "Remote data visualization through WebSockets," in *Proceedings of the Eighth International Conference on Information Technology: New Generations IEEE*, pp. 1050-1051, Las Vegas, NV, USA, April 2011.

[21] L. S. Huang, E. Y. Chen, A. Barth, E. Rescorla, and C. Jackson, "Talking to yourself for fun and profit," in *Proceedings of the W2SP*, pp. 1–11, Oakland, CA, USA, May 2011.

[22] J. P. Erkkilä, *Websocket Security Analysis*, pp. 2-3, Aalto University School of Science, 2012.

[23] L. Fette and A. Melnikov, *RFC 6455, "The WebSocket Protocol,"* 2011.

[24] M. Shema, S. Shekyan, and V. Toukharian, *Hacking with WebSockets*, BlackHat USA, London, UK, 2012.

[25] S. Shah, *Html5 Top 10 Threats Stealth Attacks and Silent Exploits*, BlackHat Europe, London, UK, 2012.

[26] "What is OWASP? what are the OWASP top 10?" 2020, https://www.owasp.org/index.php/Top_10_2013-Top_10.

[27] "What is OWASP? what are the OWASP top 10?" 2020, https://www.cloudflare.com/learning/security/threats/owasp-top-10/.

[28] A. Bhardwaj, V. Mangat, R. Vig, S. Halder, and M. Conti, "Distributed denial of service attacks in cloud: state-of-the-art of scientific and commercial solutions," *Computer Science Review*, vol. 39, Article ID 100332, 2021.

[29] A. Tomic, "NoSQL: a relational database using NoSQL technology," Master's thesis, USI, Valhalla, NY, USA, 2011.

[30] "W3C. 2008. HTML5-A vocabulary and associated APIs for HTML and XHTML," 2020, https://www.w3.org/TR/2008/WD-html5-20080610/single-page/.

[31] I. Hickson, *The WebSocket Protocol Draft-Hixie-the WebSocket Protocol-76*, Google. Inc, Menlo Park, CA, USA, 2010.

[32] V. Chopra, *WebSocket Essentials–Building Apps with HTML5 WebSockets*, Packt Publishing Ltd, Birmingham, UK, 2015.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9954951, 12 pages
https://doi.org/10.1155/2021/9954951

Research Article

AdaGUM: An Adaptive Graph Updating Model-Based Anomaly Detection Method for Edge Computing Environment

Xiang Yu,[1] **Chun Shan**,[2] **Jilong Bian**,[3] **Xianfei Yang**,[1] **Ying Chen**,[1] **and Haifeng Song**[1]

[1]*School of Electronics and Information Engineering, Taizhou University, Taizhou 318000, China*
[2]*School of Electronics and Information, Guangdong Polytechnic Normal University, Guangdong 510006, China*
[3]*School of Information and Computer Engineering, Northeast Forestry University, Harbin 150001, China*

Correspondence should be addressed to Xiang Yu; yuxiang@tzc.edu.cn, Chun Shan; shanchun@gpnu.edu.cn, and Jilong Bian; bianjilong@nefu.edu.cn

Received 1 April 2021; Accepted 3 May 2021; Published 15 May 2021

Academic Editor: Muhammad Shafiq

With the rapid development of Internet of Things (IoT), massive sensor data are being generated by the sensors deployed everywhere at an unprecedented rate. As the number of Internet of Things devices is estimated to grow to 25 billion by 2021, when facing the explicit or implicit anomalies in the real-time sensor data collected from Internet of Things devices, it is necessary to develop an effective and efficient anomaly detection method for IoT devices. Recent advances in the edge computing have significant impacts on the solution of anomaly detection in IoT. In this study, an adaptive graph updating model is first presented, based on which a novel anomaly detection method for edge computing environment is then proposed. At the cloud center, the unknown patterns are classified by a deep leaning model, based on the classification results, the feature graphs are updated periodically, and the classification results are constantly transmitted to each edge node where a cache is employed to keep the newly emerging anomalies or normal patterns temporarily until the edge node receives a newly updated feature graph. Finally, a series of comparison experiments are conducted to demonstrate the effectiveness of the proposed anomaly detection method for edge computing. And the results show that the proposed method can detect the anomalies in the real-time sensor data efficiently and accurately. More than that, the proposed method performs well when there exist newly emerging patterns, no matter they are anomalous or normal.

1. Introduction

With the rapid development of Internet of Things and wireless technologies, more and more applications, which are enabled by IoT, such as smart home appliances, self-driving, and intelligent temperature control, appear in our daily life. As an indispensable component of IoT, all kinds of sensors are widely used for data collection in the tasks of various fields [1–3]. According to the research on the anomaly detection for sensor data [4], it is hard to determine the occurrence time and frequency of the anomalies. And it is inevitable that both normal data and anomalous data are involved in the massive collected sensor data. Hence, the anomaly detection in IoT scenarios brings us new challenges. On the one hand, anomalies should be detected in real-time rather than being detected at the cloud center after a period of time. For example, anomalous temperature data from the sensors deployed in a forest might represent a fire warning which means that immediate action is required. On the other hand, the accuracy of anomaly detection should be ensured because high false-positive rate will lead to frequent false alarm and high false-negative rate will lead to anomalies undetected.

In most traditional anomaly detection methods, a behavior is considered as anomalous when its features are the same as or similar to the features of a known anomalous pattern. First, a behavior is represented as feature vector that consists of part or all features of the behavior. Similarly, all the known patterns, either anomalous or normal, are represented as feature vectors. Second, the similarity between the behavior to be detected and each known anomalous pattern is calculated, and the behavior is considered as anomalous when the similarity exceeds a predefined threshold. Generally, the anomaly detection methods can be roughly classified as distance-based detection methods, statistics-based detection methods, density-based methods, or the methods based on neural network [5–8]. However, almost all the traditional anomaly detection methods suffer from the disadvantage of over reliance on the pretrained static detection model, which will lead to the degradation of detection performance when the anomalies of new type occur. In view of this, more and more machine learning algorithms are employed in anomaly detection tasks due to the characteristics of improving the detection model adaptively according to the incremental learning results [9–11]. Nevertheless, due to the requirement of real-time detection, the anomaly detection methods, in which corresponding machine learning algorithms are employed, still cannot detect anomalies from massive sensor data efficiently.

According to the statistics, the number of IoT devices is estimated to grow to 25 billion by 2021. In order to implement the anomaly detection for sensor data in IoT, the cloud computing model is widely employed where the collected sensor data are analyzed. Actually, not only the anomaly detection model is deployed at the cloud platform but also all the sensor data are collected and transmitted to the cloud platform, so that the detection model can be conveniently improved according to the changes of the collected sensor data. However, some anomalies of certain types need to be detected in real-time, rather than being detected at the cloud platform. Suppose such a scenario, and the sensor data collected from a moving vehicle show that there exist some anomalies in the components of the vehicle which makes the vehicle in danger. Obviously, the sensor data should be detected in real-time, and the corresponding measures should be taken immediately so as to avoid the occurrence of an accident. In addition, with the growing scale of IoT, more and more wireless devices are involved, which means an increase of the computing pressure on cloud platform [12, 13]. In order to alleviate the pressure on both the network bandwidth and the computing power of cloud platform, a possible solution is the edge computing, which migrates computation-intensive tasks from the cloud platform to the edge severs where the delay-sensitive detection tasks can also be performed in real-time.

The remainder of the study is organized as follows. First, the related work on anomaly detection is introduced. Second, we propose an adaptive graph updating model (Ada-GUM), which records the features of each known patterns. Third, the architecture that can dynamically orchestrate the anomaly detection method for edge computing and the feature graph updating model is described in detail. Fourth, several experiments are conducted in order to validate both the efficiency and the effectiveness of the proposed method. Finally, we conclude the study and discuss the research focus of our future work.

2. Related Work

2.1. Traditional Anomaly Detection Methods. The basic principle of anomaly detection is to detect the data which are deviated from the known normal patterns according to a predefined similarity threshold. Currently, the common anomaly detection methods include the anomaly detection methods based on distance [14, 15], the anomaly detection methods based on density [16, 17], the anomaly detection methods based on classification [18, 19], and the anomaly detection methods based on models [20–22]. With the development of machine learning, lots of advanced algorithms and models have been proposed successively and integrated with the existing traditional anomaly detection methods in order to improve their detection performance on both accuracy and efficiency [23–25], such as the detection methods based on the neural network model, the detection methods based on k nearest neighbor (k-NN), and the detection methods based on the support vector machine (SVM).

After years of research, lots of anomaly detection methods for different application scenarios have been proposed in succession. In year 1994, based on a statistical model, Barnett et al. proposed an anomaly detection method, whereby the data different from the given distribution are detected as outliers. In year 1999, a classic anomaly detection method, which is based on the local outlier factor (LOF), is first proposed by Breunig et al. After that, based on the LOF, a series of similar anomaly detection methods are proposed in succession [26–29]. In year 2000, based on k-NN, Ramaswamy et al. proposed a detection method whereby anomalies can be detected without any prior knowledge [30]. In year 2001, an anomaly detection method based on the support vector machine (SVM) is proposed by Schlkopf et al. [31]. In year 2009, Angiulli et al. propose a novel algorithm which can detect outliers from very large datasets [32]. In year 2011, on the basis of several different similarity measurements, Moshtaghi et al. propose an anomaly detection method [33]. In year 2014, according to the calculation results of top-k distance, Shaikh et al. proposed an anomaly detection method [34]. In year 2014, Shaikh et al. proposed a method for anomaly detection, which can detect outliers from uncertain datasets [1]. In year 2014, Huang et al. proposed an anomaly detection model, and different machine learning algorithms can be separately integrated with the proposed model [35]. However, with the emergency of more and more big data applications, traditional anomaly detection methods might suffer from the following disadvantages in the big data processing tasks. First, the performance of some traditional methods fluctuates with the data distribution. Second, the performance of some traditional methods deteriorates with the increase of data dimension. Third, the performance of some traditional methods is limited by the predetermined models. In addition, the performance of some traditional methods depends too much on the prior knowledge.

2.2. Anomaly Detection for Edge Computing. With the development of the technologies of Internet of Things and wireless sensor networks (WSN), when analyzing the massive sensor data, traditional methods become more and more inapplicable. Evidently, due to the resource constraints of each sensor node, it is impracticable for traditional methods to detect anomalies from the collected data at each sensor node under the real-time requirements. In view of the existing problem, some promising anomaly detection methods for IoT devices have been proposed in succession. And it should be noted that most existing anomaly detection methods for the sensor data from IoT devices are based on the cloud computing model and big data processing platform [36, 37]. With the powerful computing and analyzing power of the cloud computing center, the data collected from IoT devices are transmitted to the cloud computing center for further detection. However, as estimated, the IoT devices will grow to 25 billion by 2021, and it is impracticable to transmit all the data, which are collected from IoT devices, to the cloud computing center due to the limitation of network bandwidth; moreover, it is also impracticable to process or analyze the massive collected data at the cloud center due to the limitation of computing power.

As the considerable transmission cost and the pressure on computing power may degrade the performance of anomaly detection, especially for the computing-intensive and delay-sensitive tasks, some anomaly detection methods based on the edge computing strategy are proposed, whereby some computing tasks are migrated from the cloud center to network edge devices (nodes). In year 2017, Cai et al. integrated the strategy of distributed recursive computing with a selection method based on k-NN, so as to detect anomalies from time series data [38]. In year 2018, in order to detect anomalies from sensor data in real-time, a novel anomaly detection method is proposed by Zhang et al. [39]. In year 2020, Anand et al. proposed an edge-based intrusion detection method for IoT devices, which can distinguish evolving forms of attacks from normal behaviors [40]. Additionally, some research studies on anomaly detection methods based on the deep learning model are proposed in succession whose performance is better than that of the aforementioned methods based on shallow learning models [41, 42]. Although most existing anomaly detection methods for edge computing can effectively achieve service decentralization and computing tasks migration, they still suffer from the detection accuracy, especially when there exist the sensor data of some unknown patterns that cannot be identified by the pretrained static model.

In this study, we propose AdaGUM, an adaptive graph updating model, based on which a novel anomaly detection method for edge computing is proposed. From the perspective of edge nodes, anomalies can be detected on each edge node in real-time according to the feature graph which is preserved and periodically updated. From the perspective of cloud computing center, the collected data of some unknown patterns can be first identified by a deep learning model, and then, the identified patterns are not only transmitted to the cache of each edge node constantly but also stored on the cloud computing center for the coming periodic updating of feature graph.

3. AdaGUM Model

In this section, according to the formulation of anomaly detection for edge computing, the related definitions are first introduced and then follows the description of the deep learning model together with the AdaGUM model for edge computing; finally, the solution of feature graph updating together with the corresponding algorithms is proposed in detail.

3.1. Problem Formulation. Suppose the data collected from different sensors, which consist both normal data and anomalies, are denoted as $D_t = \{s_1(t), s_2(t), \ldots, s_n(t)\}$, where the subscript i means that the sensor data are collected from the i^{th} sensor, and t denotes the generation time of the data. In addition, each $s_i(t)$ can be further denoted as $s_i(t) = \left\{s_{i1}(t), s_{i2}(t), \ldots, s_{ij}(t)\right\}$, where $s_{ij}(t)$ denotes the value of data $s_i(t)$ on the j^{th} dimension. And $s_{ij}(t)$ can be abbreviated as s_{ij} when the data generation time is not considered.

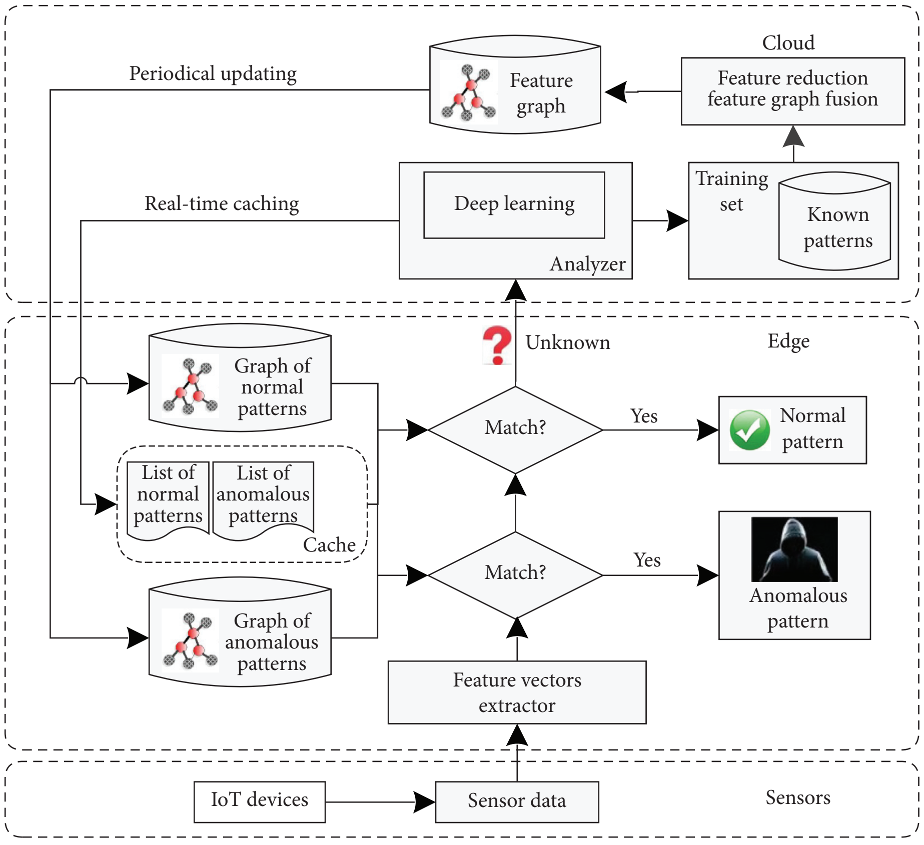
Definition 1. Anomalous value: if the values of the sensor data on one or more dimensions exceed the normal range, the values are considered as anomalous values. Actually, an anomaly may contain more than one anomalous values.

Definition 2. Anomalous pattern: an anomalous pattern consists of the corresponding anomalous range on each dimension, which can be represented as $\xi = \{\xi_1, \xi_2, \ldots, \xi_k\}$, $1 \le k \le j$, where ξ_k denotes an anomalous range on the k^{th} dimension.

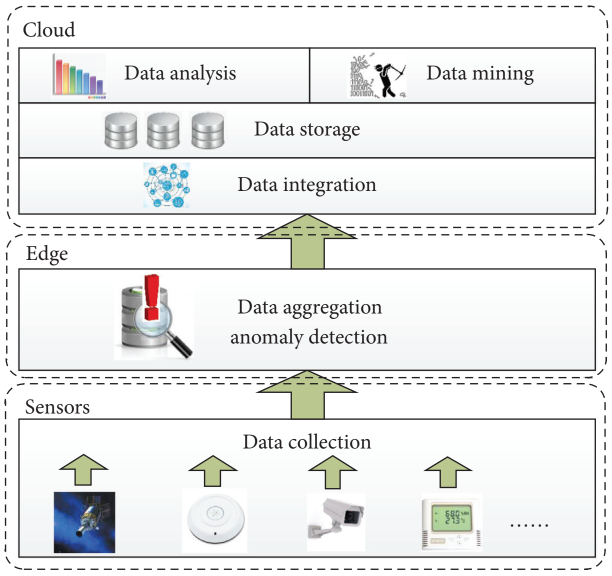
Definition 3. Anomalous feature vector: an anomalous feature vector can be represented as $\text{AVec} = \{s_{ik} | s_{ik} \notin \xi_k', 1 \le k \le j\}$, where ξ_k' denotes the normal range on the k^{th} dimension.

Definition 4. Anomalous feature graph: all the anomalous feature patterns can be integrated into a anomalous feature graph, whereby each existing anomalous feature vector corresponds to a series of nodes in the graph. Moreover, the anomalous feature graph can be updated periodically at the cloud computer center according to the feature changes of anomalies.

3.2. The AdaGUM Model for Edge Computing. As discussed before, the IoT anomaly detection solutions can be broadly classified into the centralized anomaly detection methods based on the cloud computing center and the distributed anomaly detection for edge computing. Evidently, with the increasing of IoT devices, which bring more and more sensor data, it is necessary to take the considerable transmission cost and the latency of data processing or analyzing into account. In this study, we propose the AdaGUM model for edge computing which can perform the computation-intensive and delay-sensitive anomaly detection tasks effectively. As the functional structure of AdaGUM shown in Figure 1(b), at the sensors layer, the data are first collected from various of sensors in real-time and then transmitted to the corresponding edge layer for detection. Concurrently, the emerging unknown patterns, which cannot be identified, are reported to the cloud layer for further processing. With the powerful capability of analyzing and processing, the cloud computing center plays a crucial role and achieves a more intelligent way of work. And the main functions of the cloud computing center include data storage, data integration, and data analysis.



(a)



(b)

Figure 1: The AdaGUM model. (a) The logical deployment of AdaGUM. (b) The functional structure of AdaGUM.

As the logical deployment of AdaGUM shown in Figure 1(a), the main function of the sensors layer is to collect and transmit the data generated by various emerging applications, which are enabled by the IoT and wireless technologies. Hence, we put focus on the logical functions and deployment details of the edge layer and cloud layer.

3.2.1. The Edge Layer in AdaGUM. As the edge layer shown in Figure 1(a), the sensor data to be detected are first preprocessed, if necessary, and transformed into the feature vectors by the feature vectors extractor. Next, the vectors are compared with both the list of anomalous patterns, which is temporarily kept in cache until the next periodical feature graph updating, and the graph of anomalous patterns where the existing anomalous patterns that have emerged up to the latest updating is recorded. When the similarity between the feature vector to be detected and an existing anomalous pattern exceed a certain threshold, the vector is identified as a potential anomaly. Otherwise, the feature vector is further compared with the list of normal patterns in cache and the graph of normal patterns. When the similarity between the feature vector to be detected and an existing normal pattern matches, the feature vector is considered as normal. If neither of the aforementioned conditions is satisfied, the feature vector is transmitted to the cloud computing center for further analysis.

3.2.2. The Cloud Layer in AdaGUM. As the cloud layer shown in Figure 1(a), the set of initial known patterns, including both the anomalous patterns and the normal ones, can be obtained by learning from the training set, and the corresponding feature graphs are generated based on the known patterns. During the periodical updating, the feature graphs, which record the anomalous patterns and the normal patterns, are allocated to each edges node. Next, the unknown feature vectors reported from edge nodes are analyzed and labelled by the analyzer, where a deep learning model is built to judge whether there exist anomalies or not. As a binary classification problem, the input of the model is a n-dimensional time series data, and its output is a 2-value classification which are anomalous or normal. As Figure 2 of model structure shows, each convolution module consists of 1-dimension convolution, batch normalization, and dropout.

In equation (1), S and O denote the input and output, respectively, and in equation (2), B denotes the bias, W denotes the matrix of weights, p denotes the probability, μ denotes the mean, σ denotes the variance, and ϵ is a small value which prevents the denominator from being zero. And the function of convolution is to detect the patterns in the sequence and output the corresponding features. The input of the batch normalization layer is the output of the convolution layer, and the input is normalized subsequently. Then, the negative values are filtered by the rectified linear unit, and nonlinearity is introduced. In order to avoid overfitting, part of the outputs are removed by dropout randomly. And the decision layer consists of several fully connected layer modules which are constructed by the linear layer, batch normalization, and dropout.

$$O = \mathrm{Dropout}(\mathrm{ReLU}(\mathrm{batch}(\mathrm{conv}(S)))), \tag{1}$$

$$\begin{cases} \mathrm{Conv}(S)_{k,i} = B_k + \sum_{l=1}^{m_k}\sum_{j=1}^{C} S_{i+l-1,j}W_{k,l,j}, \\ \mathrm{Batch}(X) = \dfrac{X-\mu_X}{\sqrt{\sigma_X^2+\varepsilon}}, \\ \mathrm{ReLU}(x) = \max(0,x), \\ R \sim \mathrm{Bernoulli}(p), \\ \mathrm{Dropout}(X) = R * X. \end{cases} \tag{2}$$

In equation (3), S_f is the output of feature layers, the results of classification P denotes the probability outputs that consist of two values. B denotes the bias and W denotes the matrix of weights.

$$\begin{aligned} P &= \mathrm{Dropout}(\mathrm{ReLU}(\mathrm{batch}(\mathrm{linear}(S_F)))), \\ \mathrm{Linear}(S_F) &= B + W \times S_F. \end{aligned} \tag{3}$$

After being classified, the reported feature vectors together with their labels are transmitted to the lists of both normal patterns and anomalous patterns on each edge node, respectively. Simultaneously, the classification results of the reported feature vectors are also integrated with the collection of known patterns at the cloud computing center. Subsequently, the feature graph can be updated later based on the new collection of known patterns. It is worth noting that m the sensor data transmitted to the edge nodes are detected in terms of both the periodically updated feature graphs from the cloud computing center and the lists of patterns which is constantly updated.

3.3. The Feature Graph. Generally, each sensor data generated from IoT devices can be regarded as a vector that consists of one or more values on the corresponding dimensions. As shown in Figure 3, the range of sensor data on each dimension is partitioned into different intervals denoted as $\xi_{m1}, \xi_{m2}, \ldots, \xi_{mn}$, where m denotes the serial number of the dimension, and $1, 2, \ldots, n$ denotes the serial number of the intervals on the $\mathrm{m^{th}}$ dimension. Then, each sensor data can be located in the corresponding intervals on each dimension. For example, suppose there exist sensor data $s_i = \{s_{i1}, s_{i2}, \ldots, s_{im}\}$, which is transmitted to a nearest edge server/node. First, the m-dimensional data space is partitioned into $m * n$ intervals, where each dimension consists of n intervals. Second, the components in s_i is reordered according to a predefined order, so as to obtain the corresponding feature vector. Assuming feature vector 1 in Figure 3 is the feature vector which is obtained after the reordering step. Then, feature vector 1 can be located in the corresponding interval of each dimension according to its values on all m dimensions, as the red dotted lines indicate. Similarly, all sensor data can be transformed into feature vectors. After all anomalies in the training set are located in the corresponding intervals, a feature graph of anomalous patterns can be obtained by extracting all the nonempty intervals on each dimension. Similarly, the feature graph of normal patterns can be constructed in the same way. It is worth noting that the links between intervals are preserved as the edges in feature graph.

Evidently, from the construction process of feature graph, we can draw the conclusion that a newly emerging anomalous pattern can be easily merged into the a existed feature graph. On the contrary, an old anomalous pattern can also be removed from a existed feature graph when it is no longer considered as anomalous. As shown in Figure 4, it is flexible to preserve anomalous patterns with feature graph. On the one hand, different feature graphs can be fused as needed. On the other hand, obsoleted patterns can be easily removed from feature graph.

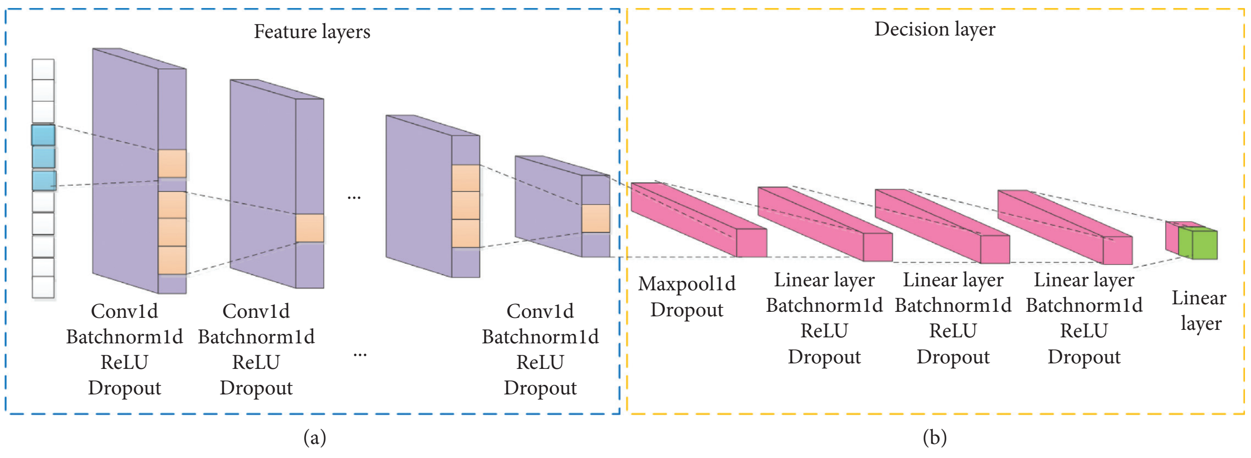


FIGURE 2: The 2-value classification model.

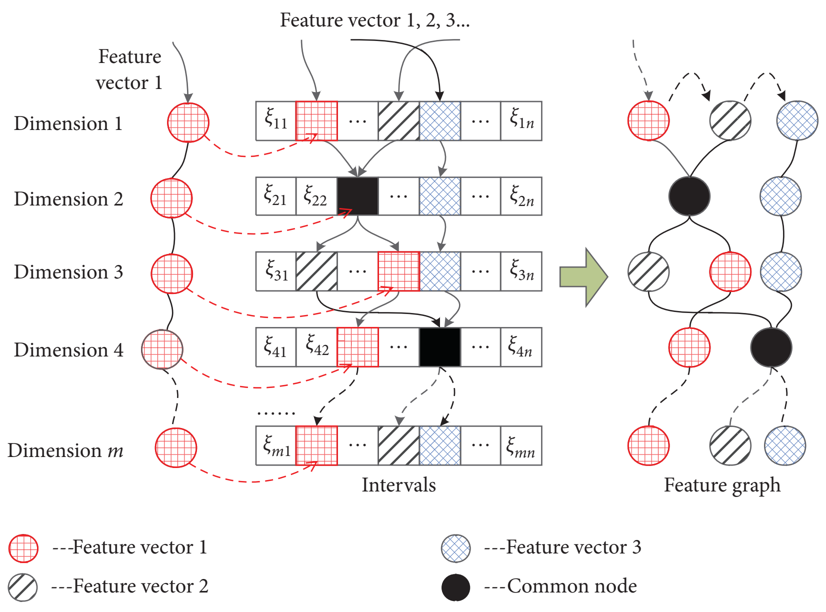


FIGURE 3: The construction of feature graph.

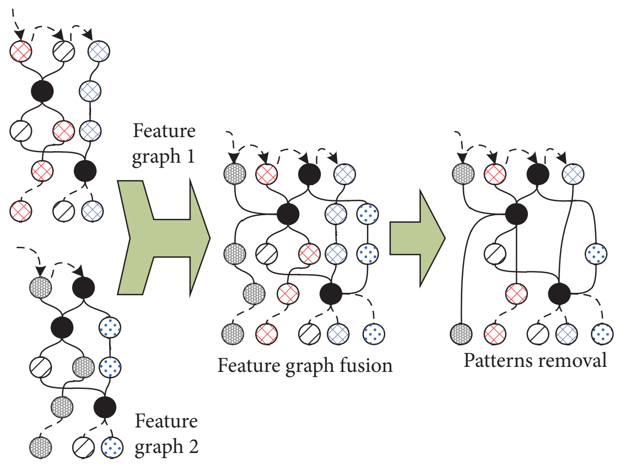


FIGURE 4: The feature graph fusion and the existed patterns removal.

3.4. Anomaly Detection Algorithm

3.4.1. Anomaly Detection on Edge Nodes. In this section, the anomaly detection algorithms on edge nodes and the cloud computing center are described in detail, respectively. Explicitly, the anomaly detection algorithm deployed on edge nodes is different from the one which is deployed at the cloud computing center. As the anomaly detection algorithm on edge nodes described in algorithm 1, sensor data are first transformed into feature vectors, and then, the vectors are compared with not only the graphs of anomalous patterns and normal patterns but also the lists of anomalous patterns and normal patterns which are preserved in cache. If the vector does not match all the patterns, it is reported to the cloud computing center for further detection.

3.4.2. Anomaly Detection at Cloud Computing Center. As the anomaly detection algorithm at the cloud computing center described in algorithm 2, with the intelligent machine learning methods, the reported unknown vectors are clustered and labelled as normal or anomalous. Subsequently, the newly emerging patterns that consist of both anomalous patterns and normal ones are distributed to the cache on each edge node. After a updating cycle, the importance of each feature in all known patterns are recalculated, whereby the feature graph is updated and distributed to all edge nodes.

4. Experiments

In this section, we evaluate the performance of the proposed anomaly detection method for edge computing through extensive simulations. First, we provide details on the hybrid datasets which include both normal data and anomalies. Second, the indexes, which are employed to evaluate the results of performance comparison between AdaGUM and the baseline methods, are introduced. And the formulas for the calculation of index values are outlined. Third, we further introduced the baseline anomaly detection methods whose performance are compared with the proposed method. Finally, we compare the performance of AdaGUM with that of the baseline methods based on the experiments which are conducted on the same datasets and analyze the experimental results in detail.

4.1. The Datasets. In this study, the experiments for the performance evaluation of AdaGUM are conducted on several representative datasets. The first one is the dataset named detection of IoT botnet attacks, which is abbreviated as BaIoT. There are 7062606 instances, each of which has 115 attributes, involved in the BaIoT dataset. Authentically, the data in BaIoT are gathered from 9 commercial IoT devices which are infected by Mirai and Bashlite. And there does not exist missing values in the data of BaIoT. The second one is the El Nino dataset whose data are collected from the Tropical Atmosphere Ocean (TAO) array, which consists of over 70 moored buoys. There are 178080 instances, each of which has 12 attributes, involved in the El Nino dataset. Since there exist missing values, the experimental data in the El Nino dataset need to be preprocessed. In addition, in order to further test the flexibility and scalability of AdaGUM, we construct a synthetic dataset with 20000 instances from the Mulcross data generator, and the proportions of known anomalies and unknown anomalies are both 5%. The basic information of the experimental datasets is given in Table 1.

4.2. The Baseline Methods. Due to the outstanding performance on clustering and classification, two classic machine learning methods, k-nearest neighbor and one-class support vector machine (SVM), are chosen as the baseline methods for the performance comparison with AdaGUM. As to the parameter settings, the best parameters for these algorithms are selected by the leave-one-out cross-validation and a grid search strategy. Moreover, based on the adaptive graph updating strategy, we develop an anomaly detection method for cloud computing, which is recorded as AdaGUM_CL in order to distinguish from the proposed anomaly detection method for edge computing, AdaGUM. Then, we compare the performance between the four methods on both the detection effects and the detection efficiency.

4.3. The Evaluation Indexes. In the experiments, we evaluate the performance of AdaGUM with two indexes which have been generally employed based on consensus, that is, true-positive rate (TPR) and false-positive rate (FPR). In the process of anomaly detection, TPR indicates the ratio of the true anomalies which are correctly detected. Evidently, the task of anomaly detection is to detect the anomalies correctly as much as possible, as to the TPR index, the higher the better. On the contrary, FPR indicates the ratio of the data, which are actually normal and mistakenly detected as anomalies. As to the FPR index, the lower the better. The calculation formulas of TPR and FPR are listed in equations (4) and (5), where true-positive (TP) denotes the anomalies which are detected correctly, false-negative (FN) denotes the anomalies which are mistakenly detected as normal data, false-positive (FP) denotes the normal data which are mistakenly detected as anomalies, and true-negative (TN) denotes the normal data which are detected correctly.

$$\text{True} - \text{positive rate}\,(\text{TPR}) = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{4}$$

$$\text{False} - \text{positive rate}\,(\text{FPR}) = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{5}$$

4.4. Experimental Results and Analysis

4.4.1. Detection without Unknown Anomalies. In this experiment, each sensor data to be detected must be in accordance with a certain known pattern. Then, we compare the performance of AdaGUM with the performance of the other three anomaly detection methods, which are

```
Input: $\theta$-the threshold of similarity degree
$G_{\text{Ano}}$-the feature graph of anomalous patterns
$G_{\text{nor}}$-the feature graph of normal patterns
$L_{\text{Ano}}$-the list of anomalous patterns in cache
$L_{\text{nor}}$-the list of normal patterns in cache
Output: $C_{\text{Ano}}$-the collection of detected anomalies
$C_{\text{Nor}}$-the collection of detected normal data
$C_{\text{Unk}}$-the collection of the data whose pattern are unknown
(1)$C_{\text{Ano}} \leftarrow \phi$
(2)$C_{\text{Unk}} \leftarrow \phi$
(3)$D \leftarrow \phi$
(4)flag←UNLABELLED
(5)Q←$\phi$;//The queue of sensor data
(6)**while** (GetSensorData$(D)$)
(7){$Q \leftarrow Q \bigcup$ Transform$(D)$;
(8)    **while** (!Empty$(Q)$)
(9)    {$s \leftarrow$ DeQueue$(Q)$;
(10)       **for** (each $p$ in $G_{\text{Ano}}$)
(11)           **if** (Match$(s, p, \theta)$)
(12)               {$C_{\text{Ano}} \leftarrow C_{\text{Ano}} \bigcup s$; flag←LABELLED; }
(13       **for** (each $p$ in $G_{\text{Nor}}$)
(14)           **if** (Match$(s, p, \theta)$)
(15)               {$C_{\text{Nor}} \leftarrow C_{\text{Nor}} \bigcup s$; flag←LABELLED; }
(16)       **for** (each $l$ in $L_{\text{Ano}}$)
(17)           **if** (Match$(s, l, \theta)$)
(18)               {$C_{\text{Ano}} \leftarrow C_{\text{Ano}} \bigcup s$; flag←LABELLED; }
(19)       **for** (each $l$ in $L_{\text{Nor}}$)
(20)           **if** (Match$(s, l, \theta)$)
(21)               {$C_{\text{Nor}} \leftarrow C_{\text{Nor}} \bigcup s$; flag←LABELLED; }
(22)           **if** (flag! = LABELLED)
(23)               {$C_{\text{Unk}} \leftarrow C_{\text{Unk}} \bigcup s$; }
(24)       flag←UNLABELLED;
(25)    }
(26)}
(27)**return**$C_{\text{Ano}}, C_{\text{Nor}}, C_{\text{Unk}}$;
```

Algorithm 1: Anomaly detection on edge nodes.

AdaGUM_CL, one-class SVM, and k-NN. The experiment is conducted on both BaIoT dataset and El Nino dataset. As to kernel of one-class SVM, the widely approved radial basis function kernel is employed. As to k-NN, we set the parameter $k = 30$, which is larger enough compared with the size of any anomaly cluster. As to AdaGUM and AdaGUM_CL, we set $\theta = 0.8$, which means that sensor data are considered to be a pattern only if more than 80% of its dimension values are in accordance with the corresponding ranges of the pattern. And the feature graph is updated once when every 1000 sensor data are detected. As to AdaGUM and AdaGUM_CL, we focus on the average TPR and FPR within each updating cycle. As the performance comparison shown in Figure 5, all the four methods perform well; however, the performance of k-NN is slightly inferior to the other three methods, which is mainly because the performance of k-NN is influenced by the initial value of k, and the best k value is hard to be determined.

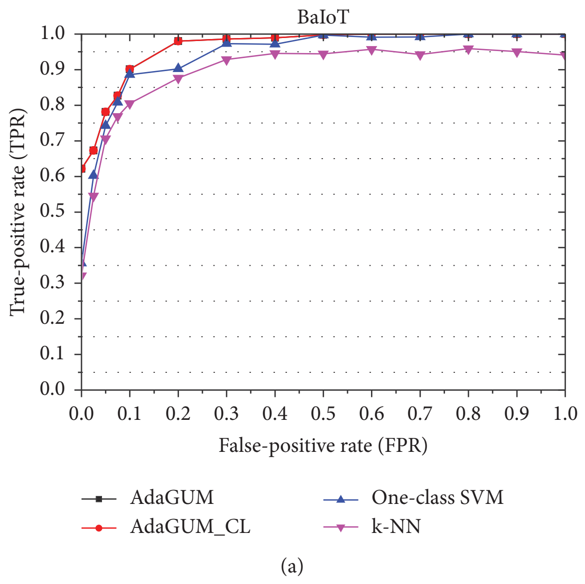
4.4.2. Detection with Unknown Anomalies. In this experiment, 10% of the known anomalies and normal data are initially considered as unknown whose labels are removed artificially. Then, we compare the performance of AdaGUM with the performance of the other three anomaly detection methods. The experiment is conducted on both BaIoT dataset and El Nino dataset. As to kernel of one-class SVM, the widely approved radial basis function kernel is employed. As to k-NN, we set the parameter $k = 30$, which is larger enough compared with the size of any anomaly cluster. As to AdaGUM and AdaGUM_CL, we set $\theta = 0.8$. And the feature graph is updated once when every 500 sensor data are detected. As to AdaGUM and AdaGUM_CL, we focus on the average TPR and FPR within each updating cycle. As the performance comparison shown in Figure 6, AdaGUM outperforms the other three methods because when there exist the data of unknown patterns, AdaGUM and AdaGUM_CL can report the data to the cloud computing center where the data can be further analyzed and processed; however, AdaGUM_CL cannot identify the newly emerging patterns within a updating cycle until the feature graphs are updated. As to the other two baseline methods, which are based on the static model generated by learning from the train set, they inevitably fail to identify the data of unknown patterns.

```
Input: $\delta$-the threshold of feature importance
$t_0$-the latest updating time
$T_I$-the time interval of updating
$C_{\text{Unk}}$-the collection of the data whose patterns are unknown
$G_{\text{Ano}}$-the feature graph of anomalous patterns
$G_{\text{Nor}}$-the feature graph of normal patterns
Output: $G_{\text{Ano}}'$-the updated graph of anomalous patterns
$G_{\text{Nor}}'$-the updated graph of normal patterns
(1) $G_{\text{Ano}}' \leftarrow \phi$;
(2) $G_{\text{Nor}}' \leftarrow \phi$;
(3) result←DeepLearning($C_{\text{Unk}}$);
(4) Distribute2Cache($G_{\text{Ano}}', G_{\text{Nor}}'$);
(5) **while** ((GetCurrentTime()-$t_0$) $\geq T_I$)
(6)     **for** (each node $n$ in $G_{\text{Ano}}$)
(7)     {**if** (Calculate($n$) $\leq \delta$)
(8)         {Remove($n$); $G_{\text{Ano}} \leftarrow G_{\text{Ano}} - n$; }
(9)     **for** (each node $n'$ in $G_{Nor}$)
(10)        **if** (Calculate($n'$) $\leq \delta$)
(11)        {Remove($n'$); $G_{\text{Ano}} \leftarrow G_{\text{Ano}} - n'$; }
(12) }
(13) $G_{\text{Ano}}' \leftarrow G_{\text{Ano}} \bigcup G_{\text{Ano}}'$;
(14) $G_{\text{Nor}}' \leftarrow G_{\text{Nor}} \bigcup G_{\text{Nor}}'$;
(15) Distribut2Edge($G_{\text{Ano}}', G_{\text{Nor}}'$);
(16) **return**$G_{\text{Ano}}', G_{\text{Nor}}'$;
```

ALGORITHM 2: Anomaly detection on edge nodes.

TABLE 1: The experimental datasets.

Datasets	Instances	Dimensions	Anomaly
BaIoT	7062606	115	Attack
El Nino	178080	12	Temperature and humility
Mulcross	20000	20	Data



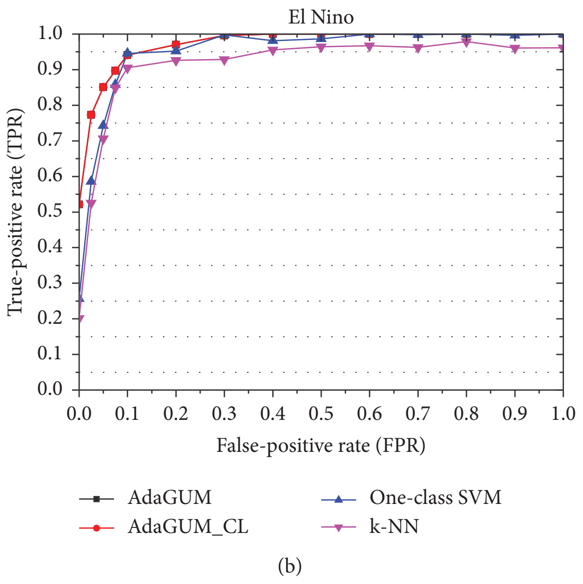
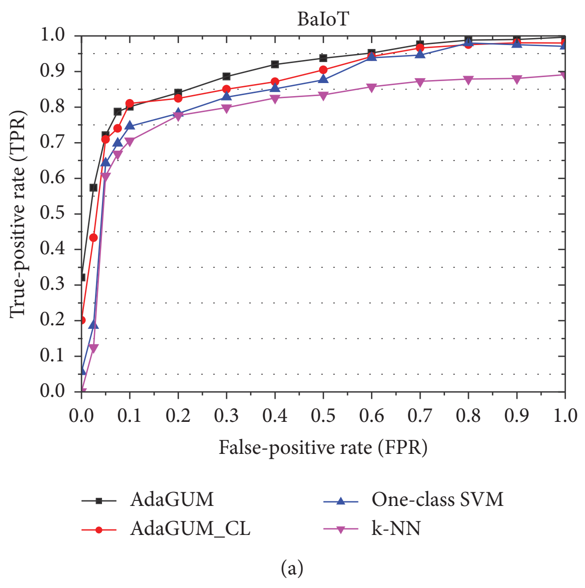


FIGURE 5: Performance comparison for anomaly detection. (a) Anomaly detection on BaIoT. (b) Anomaly detection on El Nino.

4.4.3. Comparison of the Cumulative Average Detection Time. In this experiment, the data generated by Mulcross are separated into 20 groups each of which involves 1000 data. And the cumulative average detection time of AdaGUM and AdaGUM_CL are compared. As to the parameter settings, we set $\theta = 0.8$. And the feature graph is updated once when



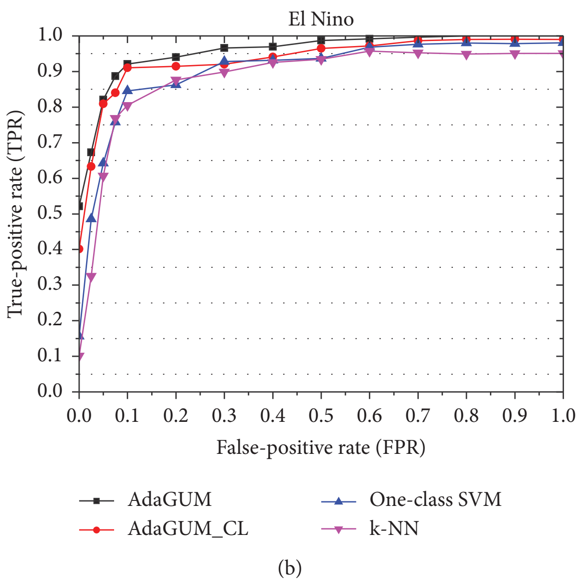


Figure 6: Performance comparison for anomaly detection. (a) Anomaly detection on BaIoT. (b) Anomaly detection on El Nino.

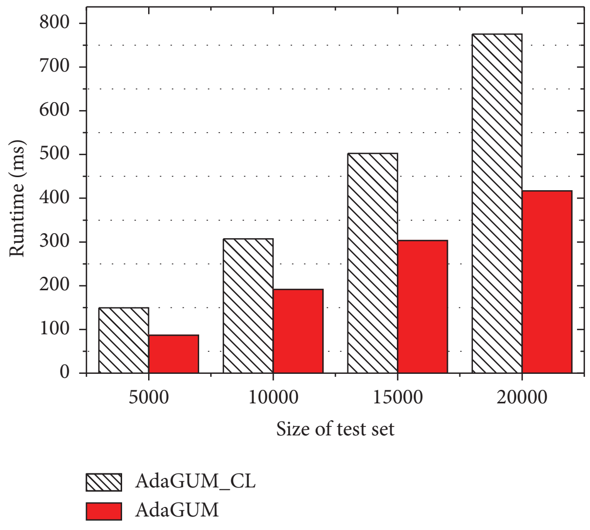


Figure 7: Comparison of total time consumption.

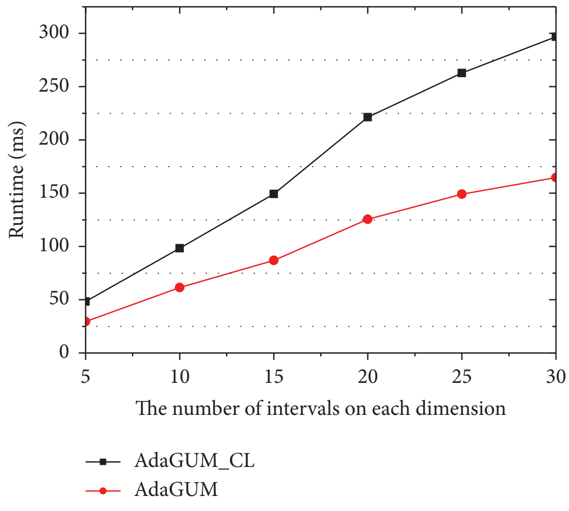


Figure 8: Comparison of total time consumption.

every 500 sensor data are detected. As the performance comparison shown in Figure 7, AdaGUM outperforms AdaGUM_CL; this is mainly because AdaGUM migrates part of the anomaly detection tasks to the edge nodes which reduces the considerable transmission cost and relatively long latency. Other than AdaGUM, AdaGUM_CL performs all the detection tasks at the cloud computing center, which degrades the detection efficiency, especially for delay-sensitive tasks.

4.4.4. Detection Performance under Different Number of Intervals. In the following experiment, the influence on the runtime of both adaGUM and adaGUM_CL, which is caused by the changes of intervals on each dimension, is tested. Then, the data generated by Mulcross are separated into 20 groups, each of which involves 500 data, and the cumulative average detection time of AdaGUM and AdaGUM_CL are compared. As to the parameter settings, we set $\theta = 0.8$. And the feature graph is updated once when every 1000 sensor data are detected. As the runtime shown in Figure 8, the runtime of AdaGUM and AdaGUM_CL both increase approximately in a linear manner, and AdaGUM still outperforms AdaGUM_CL due to its advanced detection strategy for edge computing.

5. Conclusion and Future Work

In this study, we propose AdaGUM, an anomaly detection method based on the adaptive graph updating model for edge computing. With feature graphs, both anomalous patterns and normal patterns can be explicitly recorded, and the feature graph updating strategy ensures that the features of newly emerging patterns can be integrated with the feature graph, and the features of obsoleted patterns can be removed from the feature graph. Different from the cloud computing-based anomaly detection methods, the proposed method has the following advantages:

(1) In AdaGUM, a part of the computing tasks is migrated from the cloud computing center to edge nodes where the sensor data can be detected in real-time. Then, the pressure on data processing and data transmission can be alleviated.
(2) On each edge node, the sensor data collected from IoT devices are not only compared with the known patterns recorded in the feature graph but also compared with the newly emerging patterns recorded in the lists, which ensure the detection accuracy.
(3) At the cloud computing center, with its powerful abilities of data computing and data processing, the reported unknown patterns can be analyzed and identified by intelligent machine learning methods; then, the processing results are distributed to all edge nodes which realizes the real-time detection.

Actually, in the proposed model, we focus on the co-operation between the cloud computing center and the edge node rather than the cooperations between the edge nodes. As to our future work, on the one hand, we will take the cooperations between the edge nodes into account, whereby the resources on edge nodes can be further fully utilized. On the other hand, we will try to find an efficient way of the fusion between the existing graph and a small part of newly emerged patterns.

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 (U20B2046 and 61976149), the National Key Research and Development Plan (2018YFB0803504), Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019), Heilongjiang Provincial Natural Science Foundation of China (F2018002), the Natural Science Foundation of Zhejiang Province(LZ20F020002), the Research Project of Public Technology (LGF19F020009) the Cultivating Science Foundation of Taizhou University (2019PY014 and 2019PY015), and the Science and Technology Project of Taizhou (2003gy15 and 20ny13).

References

[1] Q. Zhang, Y. Hu, C. Ji et al., "Edge computing application: real-time anomaly detection algorithm for sensing data," *Journal of Computer Research and Development*, vol. 55, no. 3, pp. 524–536, 2018.
[2] Z. Tian, W. Shi, and Y. Wang, "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.
[3] M. Shafiq, Z. Tian, A. K. Bashir et al., "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine learning techniques," *IEEE Internet of Things Journal*, vol. 57, 2020.
[4] P. Y. Chen, S. Yang, and J. A. McCann, "Distributed real-time anomaly detection in networked industrial sensing systems," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3832–3842, 2015.
[5] J. Zhao, K. Liu, W. Wang, and Y. Liu, "Adaptive fuzzy clustering based anomaly data detection in energy system of steel industry," *Information Sciences*, vol. 259, no. 3, pp. 335–345, 2014.
[6] S. A. Shaikh and H. Kitagawa, "Efficient distance-based outlier detection on uncertain datasets of Gaussian distribution," *World Wide Web*, vol. 17, no. 4, pp. 511–538, 2014.
[7] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 74–77, 2012.
[8] J. Ma, L. Sun, H. Wang et al., "Supervised anomaly detection in uncertain pseudoperiodic data streams," *ACM Transactions on Internet Technology*, vol. 16, no. 1, pp. 1–20, 2016.
[9] X. Du, "QoS routing based on multi-class nodes for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 3, pp. 241–254, 2004.
[10] F. Angiulli and F. Dolphin, "An efficient algorithm for mining distance-based outliers in very large datasets," *ACM Transactions on Knowledge Discovery from Data(TKDD)*, vol. 3, no. 1, pp. 777–781, 2009.
[11] Y. Duan, X. Fu, B. Luo et al., "Detective: automatically identify and analyze malware processes in forensic scenarios via DLLs," in *Proceedings of the IEEE ICC 2015*, London, UK, June 2015.
[12] M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, and J. Hu, "Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection," *IEEE Transactions on Information Forensics & Security*, vol. 13, no. 4, pp. 940–953, 2018.
[13] M. Zaharia, M. Chowdhury, M. J. Franklin et al., "Spark: cluster computing with working sets," *HotCloud*, vol. 15, no. 1, pp. 10–17, 2010.
[14] K. Zhang, M. Hutter, and H. Jin, "A new local distance-based outlier detection approach for scattered real-world data," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 813–822, Singapore, May 2009.
[15] W. Jin, A. K. Tung, J. Han et al., "Ranking outliers using symmetric neighborhood relationship," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, no. 1, pp. 577–593, Singapore, May 2006.
[16] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[17] J. Ma, L. Sun, H. Wang et al., "Supervised anomaly detection in uncertain pseudoperiodic data streams," *ACM Transactions on Internet Technology*, vol. 16, no. 1, pp. 1–20, 2016.
[18] H. P. Kriegel, P. Kröger, E. Schubert et al., "Dolphin: loop: local outlier probabilities," in *Proceedings of ACM Conference*

on Information and Knowledge Management, pp. 1649–1652, Hong Kong, China, November 2009.

[19] Z. Peng, P. Gurram, H. Kwon et al., "Sparse kernel learning-based feature selection for anomaly detection," *IEEE Trans on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 1698–1716, 2015.

[20] H. Huang, H. Qin, S. Yoo, and D. Yu, "Physics-based anomaly detection defined on manifold space," *ACM Transactions on Knowledge Discovery from Data*, vol. 9, no. 2, pp. 1–39, 2014.

[21] D. Mandala, F. Dai, X. Du et al., "Load balance and energy efficient data gathering in wireless sensor networks," in *Proceedings of the 1st IEEE International Workshop on Intelligent Systems Techniques for Wireless Sensor Networks, in Conjunction with IEEE MASS'06*, Vancouver, Canada, October 2006.

[22] K. Khedo, R. Doomun, and S. A. Reada, "Redundancy elimination for accurate data aggregation in wireless sensor networks," *Wireless Sensor Network*, vol. 2, no. 4, pp. 300–308, 2010.

[23] L. Cai, N. F. Thornhill, S. Kuenzel et al., "Real-time detection of power system disturbances based on k-nearest neighbor analysis," *IEEE Access*, vol. 5, no. 3, pp. 5631–5639, 2017.

[24] Y. Wang, Z. Tian, Y. Sun et al., "An IBN-based location privacy preserving scheme for IoCV," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, 2020.

[25] F. Jiang, Y. Fu, B. B. Gupta et al., "Deep learning based multichannel intelligent attack detection for data security," *IEEE Transactions on Sustainable Computing*, vol. 5, no. 2, 2018.

[26] P. Dong, X. Du, H. Zhang, and T. Xu, "A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows," in *Proceedings of the IEEE ICC 2016*, May 2016.

[27] 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.

[28] X. Du and F. Lin, "Improving sensor network performance by deploying mobile sensors," in *Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC)*, April 2005.

[29] D. Alassi and F. Alhajj, "Effectiveness of template detection on noise reduction and websites summarization," *Information Sciences*, vol. 219, pp. 41–72, 2013.

[30] S. Papadimitriou, H. Kitagawa, P. Gibbons et al., "Loci: fast outlier detection using the local correlation integral," in *Proceedings of the International Conference on Data Engineering*, no. 1, pp. 315–326, IEEE Press, Bangalore, India, March 2003.

[31] B. Schlkopf, J. Platt, J. Shawe-Taylor, and A. Smola, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[32] F. Wei, "Paradigm shift from cloud computing to fog computing and edge computing," *Journal of Nanjing University of Information Science & Technology*, vol. 8, no. 5, pp. 404–414, 2016.

[33] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2011.

[34] F. Ulah, M. Edwards, R. Ramdhany et al., "Data exfiltration: a review of external attack vectors and countermeasures," *Journal of Network and Computer Applications*, vol. 101, pp. 18–54, 2017.

[35] W. Zhang, B. Zhang, Y. Zhou, H. He, and Z. Ding, "An IoT honeynet based on multi-port honeypots for capturing IoT attacks," *IEEE Internet of Things Journal*, 2019.

[36] K. Shvacchko, H. Kuang, S. Radia et al., "The hadoop distributed file system,"in *Proceedings of the 26th Sympsium on Mass Storage Systems and Technologies(MSST),* vol. 1, pp. 1–10, IEEE, Incline Village, NV, USA, June 2010.

[37] W. Zhang, H. Wang, H. He, and P. Liu, "Detecting android malware by ORGB analysis," *IEEE Transactions on Reliability*, 2019.

[38] 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, no. 3, pp. 1963–1971, 2020.

[39] J. Qiu, Z. Tian, C. Du et al., "A survey on access control in the age of Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[40] M. Shafiq, Z. Tian, A. Bashir et al., "IoT malicious traffic identification using wrapper-based feature selection mechanisms," *Computers & Security*, vol. 94, Article ID 101863, 2020.

[41] V. Bui, V. H. Nguyen, T. L. Pham et al., "RNN-based Deep Learning for One-Hour Ahead Load Forecasting," in *Proceedings of the 20202 International Conference on Artificial Intelligence in Information and Communication(ICAIIC)*, pp. 587–589, IEEE, Fukuoka, Japan, April 2020.

[42] W. Kong, Z. Y. Dong, Y. Jia et al., "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, 2017.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5521713, 9 pages
https://doi.org/10.1155/2021/5521713

Research Article

TBSMR: A Trust-Based Secure Multipath Routing Protocol for Enhancing the QoS of the Mobile Ad Hoc Network

Mohammad Sirajuddin,[1] Ch. Rupa,[2] Celestine Iwendi,[3] and Cresantus Biamba[4]

[1]*Department of C. S. E., KHIT, JNTUK, Kakinada 522019, India*
[2]*Department of C. S. E., V. R. Siddhartha Engineering College, Vijayawada 520007, India*
[3]*Department of Computer Science, Coal City University, Enugu 400231, Nigeria*
[4]*Department of Educational Sciences, Faculty of Education and Business Studies, University of Gavle, 80176 Gavle, Sweden*

Correspondence should be addressed to Cresantus Biamba; cresantus.biamba@hig.se

Received 27 February 2021; Revised 28 March 2021; Accepted 8 April 2021; Published 21 April 2021

Academic Editor: Muhammad Shafiq

Mobile ad hoc network (MANET) is a miscellany of versatile nodes that communicate without any fixed physical framework. MANETs gained popularity due to various notable features like dynamic topology, rapid setup, multihop data transmission, and so on. These prominent features make MANETs suitable for many real-time applications like environmental monitoring, disaster management, and covert and combat operations. Moreover, MANETs can also be integrated with emerging technologies like cloud computing, IoT, and machine learning algorithms to achieve the vision of Industry 4.0. All MANET-based sensitive real-time applications require secure and reliable data transmission that must meet the required QoS. In MANET, achieving secure and energy-efficient data transmission is a challenging task. To accomplish such challenging objectives, it is necessary to design a secure routing protocol that enhances the MANET's QoS. In this paper, we proposed a trust-based multipath routing protocol called TBSMR to enhance the MANET's overall performance. The main strength of the proposed protocol is that it considers multiple factors like congestion control, packet loss reduction, malicious node detection, and secure data transmission to intensify the MANET's QoS. The performance of the proposed protocol is analyzed through the simulation in NS2. Our simulation results justify that the proposed routing protocol exhibits superior performance than the existing approaches.

1. Introduction

Mobile ad hoc network (MANET) is an assortment of mobile nodes that communicate without any fixed physical framework. MANETs have many striking features like varying topology, rapid setup, and multihop wireless communication. All these features make MANET suitable for various time-sensitive applications [1, 2]. Ad hoc network renders a promising communication facility where physical infrastructure is difficult to establish. Furthermore, MANETs allow mobile nodes to exchange information without any physical framework and administrative activities. Hence, these networks are dynamic, self-organized, and autoconfigured, allowing nodes to move arbitrarily during communication. MANETs also play an essential role in Industry 4.0. These networks can be extended and integrated with emerging technologies like cloud technologies, IoT, and machine learning techniques to develop smart applications for automating industrial needs [3, 4]. The structure of the MANET is depicted in Figure 1. Implementation of a secure routing protocol by ensuring QoS is a challenging task in the MANET because of its dynamic topology [5–7]. MANET facilitates open communication infrastructure through which any versatile node can easily join the network and participate in data transmission [8]. This open communication infrastructure provokes security breaches in the MANET [9].

Consequently, the secure and reliable routing in such a network is difficult to accomplish. Generally, routing protocols in MANETs are classified into three categories

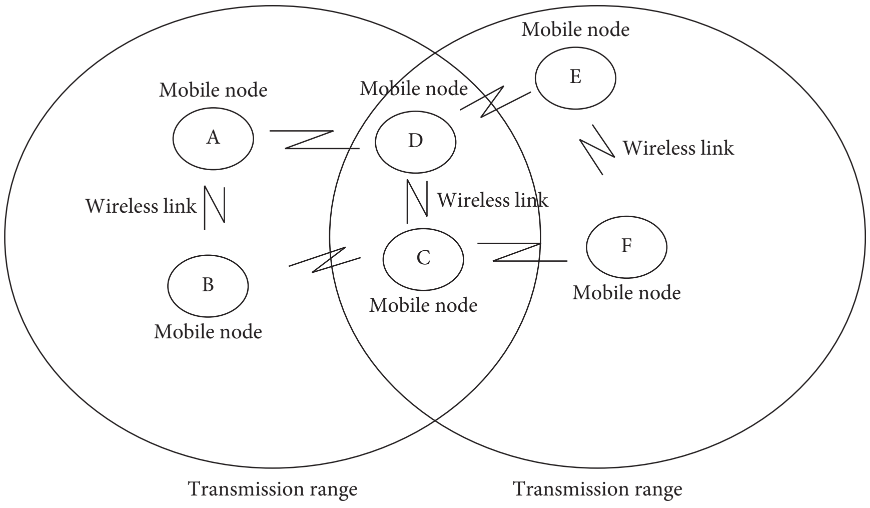


Figure 1: Structure of the MANET.

based on their design and routing process: (i) proactive routing protocols, (ii) reactive routing protocols, and (iii) hybrid routing protocols. The proactive routing protocol establishes and maintains all the routes in a routing table prior to the communication. In this category of routing protocols, route setup and route maintenance tasks are accomplished by periodically exchanging the control packets. Transmission of these control packets for route establishment and maintenance leads to routing overhead in the MANET. These proactive routing protocols are suitable for miniature networks.

Unlike proactive protocols, reactive routing protocols establish routes whenever the nodes require them. This route establishment process reduces the network overhead that occurs due to the periodic exchange of control packets in proactive routing protocols. Reactive protocols can determine an optimal path with low packet delay and network overhead compared to the proactive routing protocols. Furthermore, reactive routing protocols apply to more extensive networks. Another category of routing protocols includes hybrid routing protocols. These protocols commingle the benefits of both proactive and reactive routing protocols.

Ad hoc on-demand distance vector routing protocol (AODV) is a predominantly used reactive routing protocol in the MANET. Many researchers introspected AODV's performance by considering multiple factors and also identified various reasons that cause security breaches. This protocol has the following flaws:

(i) There is no mechanism to handle congestion.
(ii) The existing protocol does not support multipath routing.
(iii) It is susceptible to various security attacks [10].
(iv) It does not have any predefined mechanism to handle packet losses.
(v) It does not have any mechanisms to ensure QoS.
(vi) There is absence of power optimization concept.

AODV is highly susceptible to various attacks like black hole attack, wormhole attack, DoS attack, and so on. To resolve these security breaches, it is necessary to upgrade the AODV protocol.

Many researchers have proposed different flavors of AODV protocol to handle the issues mentioned above. But no versions of AODV protocols handle all the issues discussed above together as a single protocol. Therefore, the main objective of this proposed protocol is to provide a secure and efficient routing by reducing the packet losses and thereby enhancing the QoS in the MANET. In this paper, we proposed a TBSMR protocol to perform routing by considering the following factors for intensifying the network's efficiency:

(i) Routing by handling congestion.
(ii) Secure routing through trusted nodes.
(iii) Multipath routing.
(iv) Packet loss reduction.

In MANET, the fundamental reasons for packet loss are the presence of noxious nodes and lack of sufficient battery power of the nodes [11].

The proposed TBSMR protocol integrates all the properties as mentioned above for enhancing the QoS in the MANET.

2. Related Work

2.1. MANET in Industry 4.0. MANETs are in extensive use for achieving the vision of Industry 4.0. Many smarter applications based on MANETs are popping up in various domains to automate and monitor the activities. Such networks are required especially for disaster situations like earthquake, hurricane, flooding, and cyclone, to transfer emergency information for saving lives and properties. These networks can be formulated when there are lean possibilities of physical communication infrastructure. These networks are required to connect people and relay information in emergency situations as in case of recent bushfire in Australia. MANET-based real-time applications demand precise time-sensitive data transmission. Also in emergency situations, safe sheltering points and escape routes must be accurately delivered to the people in a timely manner. Many existing routing protocols do not emphasize these constraints. Furthermore, the existing routing techniques are based on low-level parameters like; delay, routing overhead, bandwidth, and so on.

Many researchers have proposed QoI-based source selection schemes to transfer time-sensitive critical information. Arsalaan et al. [12] propounded a low overhead source selection approach and QoIT that explicitly considers the user requirements to determine an optimal source, to allow information exchange in emergency situations, by avoiding bottleneck issue.

Convergence of MANET with IoT enlightens another possibility of research in smart ubiquitous computing where ad hoc network plays a vital role in implementing smart city applications. Smart city applications integrate different types of applications of various domains that require different types of message exchanges. Routing in such applications is a challenging task because of the diversity of nodes and disparate message structures. Intelligent routing techniques are required to meet the challenges of Industry 4.0. Intelligent routing protocols can be developed by using emerging technologies like machine learning, bioinspired optimization algorithms, soft computing, and so on.

3. Existing QoS-Based Routing

AODV is a conspicuously used reactive protocol in MANET. But this protocol exhibits many flaws, which are mentioned in Section 2. To overcome these flaws, many researchers developed many AODV-based routing protocols for intensifying the network's efficiency. This section emphasizes the recent flavors of AODV protocols propounded for enhancing the QoS.

Bhagyalakshmi et al. [13] proposed a Q-AODV protocol to determine a noncongested route based on the queue vacancy parameter. This queue vacancy parameter is used to reduce the number of intermediate nodes participating in the route exposure state, thereby reducing control packets' transmission.

Sarkar et al. [14] proposed an enhanced Ant-AODV protocol for optimal route selection in MANET. This protocol uses the ant colony optimization concept for the selection of optimal routes. In this technique, routing is done by computing the pheromone values of all the available paths. A path having the highest pheromone values will be used for transmitting packets from source to destination.

Jhajj et al. [15] propounded the EMAODV protocol for handling congestion. This protocol makes use of the TTL parameter to avoid the flooding of RREQ packets. This TTL parameter is used for identifying the active nodes for forwarding the packets. Only these active nodes are used for forwarding the packets. Unlike active nodes, the other nodes that do not respond to RREQ packets will be treated as silent nodes and are not involved in routing them.

Subramanian et al. [16] developed trust-based AODV in which packets are sent through trusted nodes. A node whose trust value is greater than the threshold value is treated as a trustworthy node; otherwise, it is considered an untrustworthy node. In this protocol, trust values are determined based on the number of request packets, reply packets, and data packets forwarded by each node.

Zhaoxiao et al. [17] proposed an energy-aware EAODV protocol in which a path with low energy cost and having a larger capacity is selected for data transmission. This protocol uses a priority weight parameter to predict the nodes' lifetime based on the present network traffic.

Table 1 outlines the recent existing protocols along with the properties considered to accomplish QoS routing. In our literature study, we considered recent AODV-based protocols developed to overcome the flaws of AODV protocol and we perceived that many researchers considered only a few specific aspects in extending the AODV protocol for enhancing the efficiency of the network. In the implementation of the proposed system, we considered diverse factors like congestion control, malicious node detection, packet loss reduction, and available battery power of nodes during packet transmission. We contemplated all these factors for implementing the proposed TBSMR protocol through which the QoS can be enhanced.

4. Proposed Methodology

The proposed TBSMR protocol functions in three phases for reliable packet transmissions. The three phases of the TBSMR protocol are as follows:

(i) Route exposure phase.
(ii) The malicious node detection phase.
(iii) Information forwarding phase.

This proposed TBSMR protocol is an amended version of the AODV protocol. TBSMR protocol overcomes the flaws of the AODV protocol. In the TBSMR protocol, malicious nodes are detected at every stage in communication. Moreover, the packet loss reduction mechanism is also used for reliable packet delivery. In this protocol, a spurious RREQ is broadcasted by the source node during the initial route revelation process. This spurious RREQ packet contains a fake destination address and destination sequence number. For this spurious RREQ packet, only a malicious node will respond with the RREP packet by claiming that it is having an optimal route to the destination [18, 19]. In this way, the source node identifies the malicious node based on the invalid RREPs received. After identifying the malicious node, the source node propagates this information to all other nodes so that the malicious node will not be considered for forwarding packets and detached from the network. In this way, malicious node detection and elimination are done at the earlier stages of communication. All the nodes other than the malicious nodes are treated as trusted or trustworthy nodes. Also, during communication, malicious nodes are detected and eliminated by computing the nodes' trust values. If the trust value of a node is less than

Table 1: Properties of existing routing protocols.

Protocol	Congestion control	Malicious node detection	Packet loss reduction	Energy-aware routing
Q-AODV	✓	✗	✗	✗
Enhanced Ant-AODV	✓	✗	✗	✗
EMAODV	✓	✗	✗	✗
Trust-based AODV	✗	✓	✓	✗
EAODV	✗	✗	✗	✓

the threshold trust value T_{thresh}, then that particular node is marked as a malicious node. A trust value of a node Y is computed based on the trust opinion of its neighbors. Suppose that node X is a neighbor of node Y. Node X can determine the trust opinion on node Y by using a function T (X, Y), where $T\,(X, Y)$ is a function of three parameters and it is mathematically expressed as $T\,(X, Y) = f\,[P\,(X, Y), N\,(X, Y), U\,(X, Y)]$, where $P\,(X, Y)$ represents successful packet transmissions from node X to node Y; $N\,(X, Y)$ indicates failure packet transmissions from node X to node Y; $U\,(X, Y)$ represents uncertainty factor that is initially set to 1. The uncertainty factor of value 1 represents that node X is not certain about the trustworthiness of node Y. Depending on the subsequent successful or failure packet transmissions from node X to node Y, $U\,(X, Y)$ will be updated [18]. $T\,(X, Y)$ is an average of three parameters and usually ranges from 0 to 1. The obtained value of $T\,(X, Y)$ represents node X's trust opinion on node Y and will be maintained by node X in its routing table as ***trust_val*** of node Y. For a node to be trusted, its *trust_val* $\geq = 0.6$. Every node in a network shares trust values of its neighbor nodes with all other nodes periodically, such that only trusted nodes are involved in information exchange while all the nodes whose trust_val< 0.6 will be considered as malicious and eliminated from the network. The trust value of a node is updated based on the number of packets forwarded or discarded by it on behalf of other nodes. This protocol is highly reliable because it supports the transmission of acknowledgment after successful transmission of packets from source to destination. When the destination node receives all the packets from the source node, it sends DR (Data Received) packet to the source node. After receiving the DR packet from the destination node, the source node marks the entire route as trusted through which it has transmitted data packets and received the DR packet successfully. Trust value of a route is expressed as follows: Trust (route) = successful packet transmissions/total packets transmitted.

For a route to be trusted, *Trust (route) value* ≥ 0.6.

5. Routing by Handling Congestion

This protocol supports the concept of multipath routing. This protocol allows source node to maintain multiple routes to the destination in a cache. These multiple routes are used by the source node on occurrence of congestion or link errors. In this proposed protocol, the destination node is allowed to receive multiple RREQs from the same source node for which the destination node sends multiple RREPs. On receiving multiple RREPs from the destination node, the source node selects the best route based on the number of hops for forwarding the packets. The alternate route towards destination will be stored in the sender's cache used in the future on the occurrence of congestion or link failure. The storage of an alternate path to the destination in a cache avoids invocation of the route discovery process and avoids overwhelming the selected route with packets. In existing protocols, during the routing process shortest and optimal path is opted for sending all the packets, which may lead to the congestion in the selected optimal route. Hence, unnecessary packet loss will occur, which results in reduced throughput. To address this issue, our proposed protocol is implemented so that whenever a selected optimal route is about to get congested, an immediately alternate path stored in the cache will be used by the sender for forwarding the subsequent packets. In this way, load distribution is done by determining the status of congestion of each route.

In this protocol, every node periodically sends the status of congestion to its neighbor using a QS (Queue Status) field in the HELLO packet. Each node determines the status of its available avg. queue by using the following equation:

$$\begin{aligned} \text{Min}_{\text{thresh}} &= 0.25 * \text{Total\_Buffer\_Size}, \\ \text{Max}_{\text{thresh}} &= 3 * \text{Min}_{\text{thresh}}, \\ Avg\text{\_queuenew} &= (1 - Wq) * Avg\text{\_queueold} \\ &\quad + \text{Instant\_queue} * Wq. \end{aligned} \tag{1}$$

Here, Wq is the queue weight and is a constant ($Wq = 0.002$ from RED, Floyd, 1997) *and* Instant_queue is instantaneous queue size [1].

$$\text{QS} = \text{Instant\_queue} - Avg\text{\_queuenew}. \tag{2}$$

If Queue_Status < Minthresh indicates no congestion.

If Queue_Status > Minthresh and&Instant_queue < Minthresh indicates likely to be congested.

If Instant_queue > Minthresh, indicates congestion.

Based on the above calculations, QS field is set to either 0 or 1. This QS field is set in the HELLO packet and sent periodically by each node to its neighbors. Also, before transmitting a packet, every node checks the status of congestion of neighbor nodes by transmitting special acknowledgment packets.

6. Minimizing Packet Loss

A node may discard packets intentionally or due to lack of sufficient battery power for forwarding the packets. These two cases result in unnecessary packet losses, leading to the degradation of network throughput [20–29].

Case 1. Malicious node intentionally discards the packets. A node is considered as a malicious node if the following conditions are met:

(i) Number of packets received > number of packets forwarded

(ii) Number of packets forwarded = 0

(iii) Number of packets received = number of packets forwarded and change in packet size

In all cases as mentioned above, a node is treated as a malicious node.

Case 2. Another reason for packet loss at a particular node is the lack of available battery power for forwarding the packets. In this protocol, before data transmission, the source node gets the status of its neighbor's available battery power by sending a DREQ packet. After receiving the DREQ packet, the node will send its available power to the source through the DREP packet. A node's available power or energy can be determined as follows:

Thresh_Pow = 0.10 ∗ Total_Power.where Thresh_Pow represents threshold power and Total_Power is the battery power of node.

If any node's *Avail_Pow* ≤ = *Thresh_Pow*, then this node will not be selected by the source node for forwarding the packets. In this way, a node with a lower energy level will not be considered for packet forwarding. The source node will select an alternate node with sophisticated energy for data transmission. This process will improve the packet delivery ratio through which the QoS can be enhanced.

For reliable data delivery, this proposed protocol scheme uses the concept of acknowledgment packets. Whenever the source intends to send packets to its neighbor, it sends DREQ packet to its neighbor requesting its neighbor node's status. If the neighbor node is active, it immediately responds with the DREP packet. After receiving the DREP packet, the source node sends its data. After completion of data transmission, the source node expects acknowledgment from the neighbor node. After receiving the data, the neighbor node sends DR packet to the source node. This data transmission process continues till all the packets sent by the source node reach the destination successfully. Finally, one acknowledgment packet is sent to the source from the destination node after receiving all the packets. Once the source gets an acknowledgment from the destination node, it considered the route and the intermediate nodes across the route are trusted, and at the same time, trust values are updated. In this way, the proposed scheme ensures reliable packet transmission by minimizing the packet loss; hence, the throughput of the network can be intensified.

The required step to accomplish data transmission in the proposed TBSMR is shown in Figure 2. In Figure 2, node S is the source node, and it is having a route to the destination node *D* via intermediate node A. Before sending data packets, the source node first checks the trust value of node A. If node A is a trusted node, it checks node A's status by sending the DREQ packet. On receiving DREQ, if node A is having sufficient power for forwarding the packets and is not congested, node A will immediately respond with DREP. In this way, packets are transmitted by considering nodes' trust values, congestion status, and sophisticated energy availability.

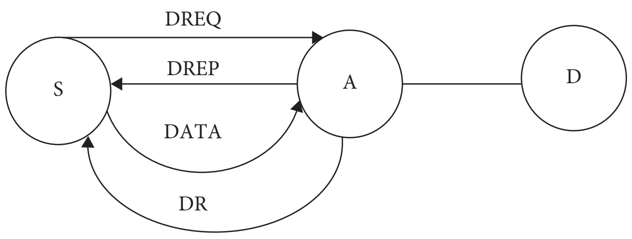


FIGURE 2: Data transmission using TBSMR protocol.

During packet transmission, a malicious node may repeatedly send DREQ packets to the source node to capture data packets. Whenever a node sends three DREQ packets consecutively, it is marked as a malicious node by the source, and this information is propagated to all other nodes in a network. The algorithm for malicious node detection is explained as follows (Algorithms 1 and 2). where nrecv is the number of received packets, nfowd represents the number of packets forwarded, and size_of_pkt means the packet size.

7. Simulation and Analysis

To analyze the performance of this proposed protocol, we used the NS2 simulation tool. Table 2 represents the parameter considered for simulation in NS2.

The formulation of the MANET by using the proposed protocol is shown in Figure 3. The proposed protocol allows communication between trusted nodes only. To evaluate the performance of the proposed protocol, we considered the metrics like PDR, PLR, average end-to-end delay, and throughput.

7.1. Packet Delivery Ratio (PDR).

It is defined as the ratio between the total number of packets received by the total number of packets actually sent.

$$\mathrm{PDR} = \frac{\text{Total no.of packets received}}{\text{Total no of packets sent}}. \tag{3}$$

7.2. Packet Loss Ratio (PLR).

It represents the number of packets lost during transmission. It is the ratio between the total number of packets lost by the total number of packets received.

$$\mathrm{PLR} = \frac{\text{Total no. of packets lost}}{\text{Total no. of packets received}}. \tag{4}$$

7.3. Average End-to-End Delay.

It is defined as an average time taken by packets to reach from source to destination; this includes transfer time, propagation delay, queuing time, and processing time.

```
Step 1: Intiate_DummyTransact ( )
  { Step 1.1: Source Node broadcasts Dummy RREQ with fake Destination Address and
  Destination Sequence Number using Send_Dummy_RREQ ( ) function.
  Step 1.2: if (RREP = = received)
  Mark corresponding node as Malicious Node.
  Propagate this malicious node ID to other nodes in a network.
  }
Step 2: Route Discovery Initiated by Source Node.
Step 3: Route Establishment by considering trust values of nodes.
Step 4: Packet Transmission from Source Node to Destination
for each node on the existing route do
Check Trust value (trust_val), Congestion status and Available energy level
    If ((trust_val>0.6) andand (QS = = 0) andand (Avail_Pow > Thresh_Pow))
    {
    Send DREQ to the next node in routing table and wait for DREP
if (DREP = = received)
Send data packets to the next node and wait for DR (Data Received) packet
    if (DR = = received)
    Mark node as trusted.
Update and propagate trust value of node.
}
    else if ((turst_val>0.6)
    {
    if ((QS = = 1) || (Avail_Pow ≤ Thresh_energy))
{
    Select Alternate route from cache for forwarding the packets.
    goto Step 4.
    }
    else if (trust_val<0.6)
    {
    Marks node as malicious and remove from the routing table.
    Select alternate route if available, Otherwise
    goto Step 2.
    }
    else
    re-establish route using step 2 and repeat the process.
    }
```

Algorithm 1: Routing process.

```
Malicious_node_detection ( )
{
for each neighbor node n
do
if (nrecv > nfowd) || (nfowd = = 0)
{
mark node n as malicious. propagate this information to other nodes
}
else if (nrecv = = nfowd)
{
If (size_of_pkt! = 512 bytes)//change in packet size
{
Mark node n as malicious node
Propagate this information to other nodes
}
}
else
forward packets to node n
}
```

Algorithm 2: Malicious node detection.

Table 2: Simulation parameter.

Parameter	Values
Coverage area	500 m × 500 m
Simulation time	500 sec
No. of nodes	50,100 and 300
Traffic type	UDP-CBR
Transmission range	250 m
Packet size	512bytes
Maximum speed	20 m/s
Routing protocol	TBSMR
Mobility model	Random way point

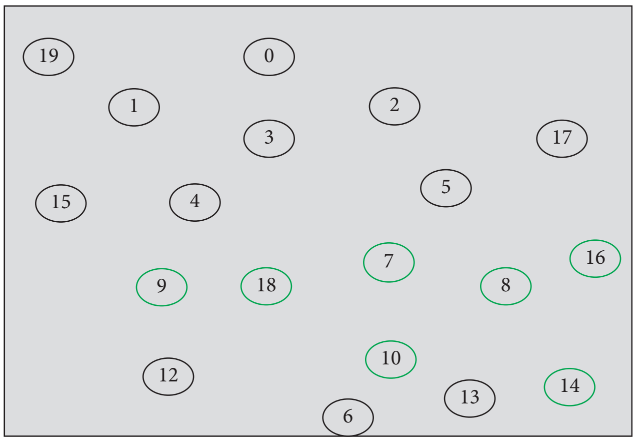


Figure 3: Formulation of MANET.

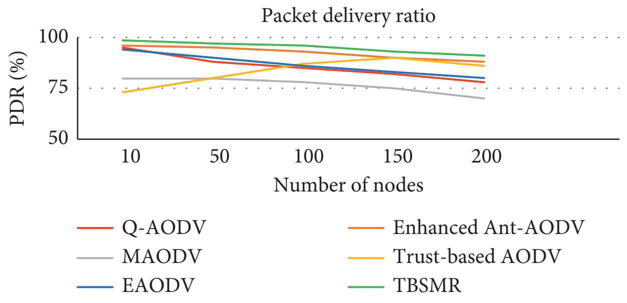


Figure 4: Comparison of PDR.

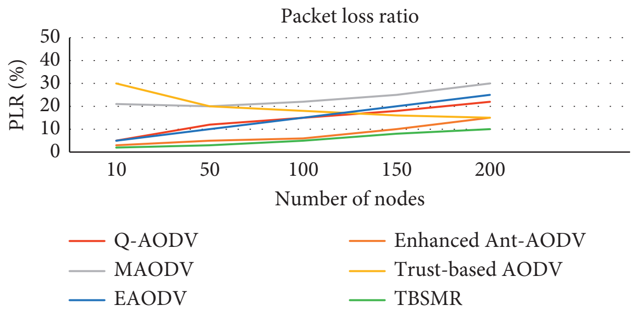


Figure 5: Comparison of PLR.

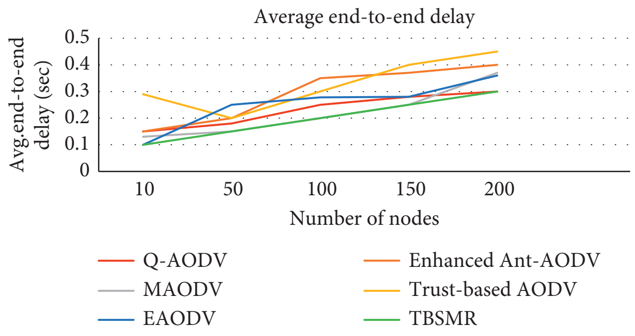


Figure 6: Comparison of average end-to-end delay.

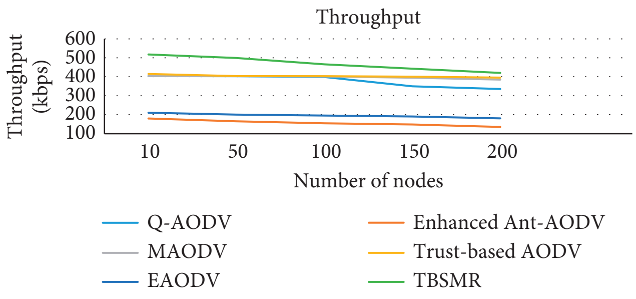


Figure 7: Comparison of throughput.

7.4. Throughput. It is the rate at which destination receives data in bits per unit time in the network. It is expressed in kbps.

Figure 4 shows that the proposed TBSMR protocol exhibits better PDR than the existing approaches.

Figure 5 illustrates that the proposed TBSMR protocol has less packet loss ratio than the existing routing techniques.

Figure 6 justifies that the proposed routing technique exhibits a lower average end-to-end delay in comparison with the other existing routing schemes.

The proposed TBSMR protocol has better throughput than the existing routing approaches considered in this study, and the same is depicted in figure 7.

Table 3 demonstrates precisely that the proposed routing technique outperforms by considering multifactors to intensify the QoS in the MANET.

Table 3: Performance comparison of the existing and proposed protocol.

Routing protocols	Congestion handling	Data loss level (%)	PDR (%)	Malicious node detection	Multipath routing	Throughput
Q-AODV	Yes	19	79	No	No	Moderate
Enhanced Ant-AODV	Yes	18	96	No	No	Low
EMAODV	Yes	20	80	No	Yes	Moderate
Trust-based AODV	No	18	91	Yes	No	High
EAODV	No	16	90	No	No	Low
Proposed method	Yes	10	98	Yes	Yes	High

All the simulated results justify that the proposed routing protocol exhibits better performance than the existing approaches in enhancing the QoS and making it suitable for real-time applications.

8. Conclusion

In this work, we proposed a routing protocol called TBSMR to enhance the QoS of the MANET. It is applicable for more extensive networks and considers multifactors like congestion, trust values of the nodes, and the available battery power of nodes during the routing process, which results in better performance with reduced overhead. Moreover, this proposed protocol supports multipath routing that minimizes the floating of unnecessary control packets for route establishment in congestion or node failure. This protocol also ensures secure communication by detecting malicious nodes. Our simulation results justify that the proposed TBSMR protocol gives better performance in PDR, PLR, average end-to-end delay, and throughput compared to the existing routing techniques. Overall, this proposed TBSMR routing approach enhances the QoS of the MANET besides ensuring secure communication.

In the future, we emphasize the implementation of security algorithms by incorporating encryption, decryption, and blockchain approaches for providing high security to the MANET.

Data Availability

Access to all the supporting data about this article will be given based on the requests directed through the data access committee and institutional review board.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] M. D. Sirajuddin, C. Rupa, and A. Prasad, "Advanced Congestion Control Techniques for MANET," *Advances in Intelligent Systems and Computing*, vol. 433, pp. 271–279, 2016.

[2] L. Femila and M. Marsaline Beno, "Optimizing transmission power and energy efficient routing protocol in MANETs," *Wireless Personal Communications*, vol. 106, no. 3, pp. 1041–1056, 2019.

[3] B. Chen, J. Wan, L. Shu, L. Peng, M. Mukherjee, and B. Yin, "Smart factory of Industry 4.0: key technologies, application case, and challenges," *Institute of Electrical and Electronics Engineers Access*, vol. 6, 2018.

[4] H. Kathiriya, A. Pandya, V. Dubay, and A. Bavarva, "State of art: energy efficient protocols for self-powered wireless sensor network in IIoT to support industry 4.0," in *Proceedings of the 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1311–1314, Noida, India, June 2020.

[5] T. Li, J. Ma, and C. Sun, "SRDPV: secure route discovery and privacy-preserving verification in MANETs," *Wireless Networks*, vol. 25, no. 4, pp. 1731–1747, 2019.

[6] T. Singh, J. Singh, and S. Sharma, "Survey of secure routing protocols in MANET," *International Journal of Mobile Network Design and Innovation*, vol. 6, no. 3, pp. 142–155, 2016.

[7] S. Hossain, M. S. Hussain, R. R. Ema, S. Dutta, S. Sarkar, and T. Islam, "Detecting Black hole attack by selecting appropriate routes for authentic message passing using SHA-3 and Diffie-Hellman algorithm in AODV and AOMDV routing protocols in MANET," in *Proceedings of the ICCCNT*, pp. 1–7, Kanpur, India, June 2019.

[8] V. V. Sarbhukan and L. Ragha, "establishing secure routing path using trust to enhance security in MANET," *Wireless Personal Communications*, vol. 110, no. 1, pp. 245–255, 2020.

[9] H. Moudni, M. Er-rouidi, H. Mouncif, and B. El Hadadi, "Secure routing protocols for mobile ad hoc networks," in *Proceedings of the IT4OD*, pp. 1–7, Fez, Morocco, March 2016.

[10] F. Abdel-Fattah, K. A. Farhan, F. H. Al-Tarawneh, and F. AlTamimi, "Security challenges and attacks in dynamic mobile ad hoc networks MANETs," in *Proceedings of the JEEIT*, pp. 28–33, Amman, Jordan, April 2019.

[11] N. Sarmah, Y. Yang, H. Sharif, and Y. Qian, "Performance analysis of MANET routing protocols by varying mobility, speed and network load," in *Proceedings of the ICSPCS*, pp. 1–6, Cairns, Australia, March 2015.

[12] A. ShakaybArsalaan and H. Nguyen, "Andrew coyle and MahrukhFida, " quality of information with minimum requirments for emergency communications," *Adhoc Networks*, vol. 111, pp. 1570–8705.

[13] Bhagyalakshmi and A. K. Dogra, "QAODV: A flood control ad-hoc on demand distance vector routing protocol," in *Proceedings of the ICSCCC*, pp. 294–299, Jalandhar, India, March 2018.

[14] D. Sarkar, S. Choudhury, and A. Majumder, "Enhanced-Ant-AODV for optimal route selection in mobile ad-hoc network," *Journal of King Saud University-Computer and Information Sciences*, vol. 8, 2018.

[15] H. Jhajj, R. Datla, and N. Wang, "Design and implementation of an efficient mul- tipath AODV routing algorithm for MANETs," in *Proceedings of the CCWC*, pp. 0527–0531, Las Vegas, NV, USA, December 2019.

[16] S. Subramanian and B. Ramachandran, "Trusted AODV for trustworthy routing in MANET," *Advances in Intelligent Systems and Computing*, vol. 167, pp. 37–45, 2012.

[17] Z. Zhaoxiao, P. Tingrui, and Z. Wenli, "Modified energy-aware AODV routing for ad hoc networks," *WRI Global Congress on Intelligent Systems*, pp. 338–342, 2009.

[18] M. S. Hussain and K. U. R. Khan, "Network-based anomaly intrusion detection system in MANETS," in *Proceedings of the ICISC*, pp. 881–886, Coimbatore, India, December 2020.

[19] M. Sirajuddin, C. Rupa, and A. Prasad, "A trusted model using improved-AODV in MANETS with packet loss reduction mechanism," *Advances in Modelling and Analysis B*, vol. 61, no. 1, pp. 15–22, 2018.

[20] M. D. Sirajuddin, C. Rupa, and A. Prasad, "An innovative security model to handle blackhole attack in MANET," *Proceedings of International Conference on Computational Intelligence and Data Engineering*, vol. 9, pp. 173–179, 2018.

[21] G. Rathee, S. Garg, G. Kaddoum, D. N. K. Jayakody, M. J. Piran, and G. Muhammad, "A Trusted Social Network Using Hypothetical Mathematical Model and Decision- Based Scheme," *Institute of Electrical and Electronics Engineers Access*, vol. 9, pp. 4223–4232, 2021.

[22] H. Lin, J. Hu, W. Xiaoding, M. F. Alhamid, and M. J. Piran, "Towards secure data fusion in industrial IoT using transfer learning," *Institute of Electrical and Electronics Engineers Transactions on Industrial Informatics*, vol. 20201 page, 2020.

[23] S. Venkatraman and M. Alazab, "Use of data visualisation for zero-day malware detection," *Security and Communication Networks*, vol. 2018, 2018.

[24] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei, and Q. Zheng, "IMCFN: image-based malware classification using fine-tuned convolutional neural network architecture," *Computer Networks*, vol. 171, Article ID 107138, 2020.

[25] Q. V. Pham, K. Dev, P. K. R. Maddikunta, T. R. Gadekallu, and T. Huynh-The, "Fusion of federated learning and industrial internet of things: a survey," 2021, https://arxiv.org/pdf/2101.00798.pdf.

[26] S. Rajadurai, M. Alazab, N. Kumar, and T. R. Gadekallu, "Latency evaluation of SDFGs on heterogeneous processors using timed automata," *Institute of Electrical and Electronics Engineers Access*, vol. 8, pp. 140171–140180, 2020.

[27] C. O. Iwendi and A. R. Allen, "Enhanced security technique for wireless sensor network nodes," in *Proceedings of the IET Conference on Wireless Sensor Systems (WSS 2012)*, pp. 1–5, London, UK, June 2012.

[28] C. Iwendi, S. Khan, J. H. Anajemba, M. Mittal, M. Alenezi, and M. Alazab, "The use of ensemble models for multiple class and binary class classification for improving intrusion detection systems," *Sensors*, vol. 20, no. 9, Article ID 2559, 2020.

[29] M. Mittal, C. Iwendi, S. Khan, and J. A. Rehman, "Analysis of security and energy efficiency for shortest route discovery in low-energy adaptive clustering hierarchy protocol using Levenberg-Marquardt neural network and gated recurrent unit for intrusion detection system," *Transactions on Emerging Telecommunications Technologies*, Article ID e3997, 2021.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 9949204, 12 pages
https://doi.org/10.1155/2021/9949204

Research Article

Compressed Wavelet Tensor Attention Capsule Network

Xiushan Liu, Chun Shan, Qin Zhang, Jun Cheng, and Peng Xu

School of Electronics and Information Engineering, Guangdong Polytechnic Normal University, Guangzhou 510665, Guangdong, China

Correspondence should be addressed to Chun Shan; shanchun@gpnu.edu.cn

Received 11 March 2021; Revised 22 March 2021; Accepted 3 April 2021; Published 17 April 2021

Academic Editor: Muhammad Shafiq

Texture classification plays an important role for various computer vision tasks. Depending upon the powerful feature extraction capability, convolutional neural network (CNN)-based texture classification methods have attracted extensive attention. However, there still exist many challenges, such as the extraction of multilevel texture features and the exploration of multidirectional relationships. To address the problem, this paper proposes the compressed wavelet tensor attention capsule network (CWTACapsNet), which integrates multiscale wavelet decomposition, tensor attention blocks, and quantization techniques into the framework of capsule neural network. Specifically, the multilevel wavelet decomposition is in charge of extracting multiscale spectral features in frequency domain; in addition, the tensor attention blocks explore the multidimensional dependencies of convolutional feature channels, and the quantization techniques make the computational storage complexities be suitable for edge computing requirements. The proposed CWTACapsNet provides an efficient way to explore spatial domain features, frequency domain features, and their dependencies which are useful for most texture classification tasks. Furthermore, CWTACapsNet benefits from quantization techniques and is suitable for edge computing applications. Experimental results on several texture datasets show that the proposed CWTACapsNet outperforms the state-of-the-art texture classification methods not only in accuracy but also in robustness.

1. Introduction

Texture classification is crucial in pattern recognition and computer vision [1–5]. Since many very sophisticated classifiers exist, the key challenge here is the development of effective features to extract from a given textured image [6]. As an important research issue, many methods have been proposed to represent texture features [7, 8]. About 51 different sets of texture features are summarized in [9]. These texture features are generally hand-crafted under some hypothesis of texture characteristics. Because different texture datasets contain different types of textures, the performance of hand-crafted features is usually changed for different datasets [4].

Recently, texture representation methods based on CNN have been achieved powerful representation capability [6, 10–13]. These CNN-based methods implement texture feature extraction in an end-to-end way which does not require predefined representation formula. Moreover, Fujieda et al. [11] find that integrating wavelet transform into CNN can effectively capture spectral information of texture images. Nevertheless, there still exist many challenges, such as extracting multilevel texture features and capturing sufficient relationships [11]. Pooling operations of CNN-based methods proactively discard substantial information which prevents the efficient exploration for texture feature relationships [14, 15]. In contrast, the capsule neural network (CapsNets), which implements dynamic routing algorithm instead of traditional pooling mechanism, can probably get rid of the weakness of pooling operations. In addition, CapsNets replaces scalar outputs of CNN with more informative vector outputs obtained by the squashing activation function. CapsNets has several advantages, such as relationship awareness and stable generalization capability [16, 17].

The attention mechanism [18] is proposed to help models to focus on more relevant regions, capture complex correlations, and discover new patterns within images.

Therefore, the integration of attention mechanism and CapsNets has great potential to represent texture features and explore their relationships sufficiently. The key problem preventing attention mechanism and CapsNets from being applied in edge computing domain is that they both suffer from heavily computation and memory burdens. It is essential to consider the quantization techniques for deploying models on edge devices [19–33]. This paper proposes the compressed wavelet tensor attention capsule network (CWTACapsNet) that integrates multilevel wavelet decomposition, tensor attention mechanism, and quantization techniques into the capsule network.

The proposed CWTACapsNet involves several compressed multiscale tensor self-attention blocks that can capture multidirectional dependencies across different channels. Furthermore, CWTACapsNet utilizes Nyström technique and proposes quantized dynamic routing process to release resource requirements. The main contributions of CWTACapsNet are three folds. First, it uses multilevel wavelet transform to extract multiscale spectral features in frequency domain which further extends texture representation capability. Second, it employs tensor attention mechanism via matrization to explore the multidirectional dependencies of texture features in different scales. Third, it employs quantization techniques to reduce the computation and memory costs without sacrificing the accuracy.

The rest of the paper is organized as follows. Section 2 presents the whole architecture and key parts of the proposed CWTACapsNet. Section 3 presents validation experiments and discusses the experimental results. The conclusion is drawn in Section 4.

2. Compressed Wavelet Tensor Attention Capsule Network

The proposed CWTACapsNet integrates multiscale wavelet decomposition and tensor self-attention blocks into capsule network. The architecture of CWTACapsNet is shown in Figure 1. CWTACapsNet involves the wavelet feature extraction block, the compressed multiscale tensor self-attention block, and the quantized capsule network. The wavelet feature extraction block extracts multiscale spectral features with multilevel wavelet decomposition. The compressed tensor self-attention block captures the multidirectional relationships within each scale, and the primary capsules are generated based on the wavelet and tensor attentive information.

2.1. Multiscale Feature Extraction via Wavelet Decomposition. Given an image $\mathbf{x}$, we utilize the 2D discrete wavelet transform (DWT) [34] with four convolutional filters, i.e., low-pass filter, $\mathbf{f}_{LL}$, and high-pass filters, $\mathbf{f}_{LH}$, $\mathbf{f}_{HL}$, and $\mathbf{f}_{HH}$, to decompose $\mathbf{x}$ into four subband images, i.e., $\mathbf{x}_{LL}$, $\mathbf{x}_{LH}$, $\mathbf{x}_{HL}$, and $\mathbf{x}_{HH}$. The convolutional stride is 2. The four filters are defined by

$$\begin{aligned}\mathbf{f}_{LL} &= \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},\\ \mathbf{f}_{LH} &= \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix},\\ \mathbf{f}_{HL} &= \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix},\\ \mathbf{f}_{LL} &= \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.\end{aligned} \tag{1}$$

The four filters (see equation (1)) indicate that they are orthogonal to each other and form a 4×4 invertible matrix. The DWT operation is given by

$$\begin{aligned}\mathbf{x}_{LL} &= (\mathbf{f}_{LL} * \mathbf{x})\downarrow_2,\\ \mathbf{x}_{LH} &= (\mathbf{f}_{LH} * \mathbf{x})\downarrow_2,\\ \mathbf{x}_{HL} &= (\mathbf{f}_{HL} * \mathbf{x})\downarrow_2,\\ \mathbf{x}_{HH} &= (\mathbf{f}_{HH} * \mathbf{x})\downarrow_2,\end{aligned} \tag{2}$$

where $*$ denotes convolution operator and $\downarrow_2$ denotes the downsampling with stride 2. The (i, j)-th value of $\mathbf{x}_{LL}, \mathbf{x}_{LH}, \mathbf{x}_{HL}$,and $\mathbf{x}_{HH}$after 2D Haar transform [19] is given by

$$\begin{aligned}\mathbf{x}_{LL}(i, j) &= \mathbf{x}(2i-1, 2j-1) + \mathbf{x}(2i-1, 2j) + \mathbf{x}(2i, 2j-1) + \mathbf{x}(2i, 2j),\\ \mathbf{x}_{LH}(i, j) &= -\mathbf{x}(2i-1, 2j-1) - \mathbf{x}(2i-1, 2j) + \mathbf{x}(2i, 2j-1) + \mathbf{x}(2i, 2j),\\ \mathbf{x}_{HL}(i, j) &= -\mathbf{x}(2i-1, 2j-1) + \mathbf{x}(2i-1, 2j) - \mathbf{x}(2i, 2j-1) + \mathbf{x}(2i, 2j),\\ \mathbf{x}_{HH}(i, j) &= \mathbf{x}(2i-1, 2j-1) - \mathbf{x}(2i-1, 2j) - \mathbf{x}(2i, 2j-1) + \mathbf{x}(2i, 2j).\end{aligned} \tag{3}$$

Based on multilevel wavelet package transform [35], the subband image $\mathbf{x}_{LL}$ is recursively decomposed by DWT. Because the downsampling stride is 2, the sizes of extracted subband images in different wavelet decomposition levels are dimidiate gradually. In addition, the upsampling operations (with stride (2) are employed to guarantee the size consistency of convolution feature maps for tensor concatenation.

2.2. Compressed Tensor Self-Attention Block. Inspired by [36], we design the *compressed tensor self-attention block* based on matricization and Nyström technique. The matricization can capture interdependencies along all dimensions of tensorized convolution feature maps. To reduce the computational and storage requirement of attention computation, we use Nyström technique to achieve an

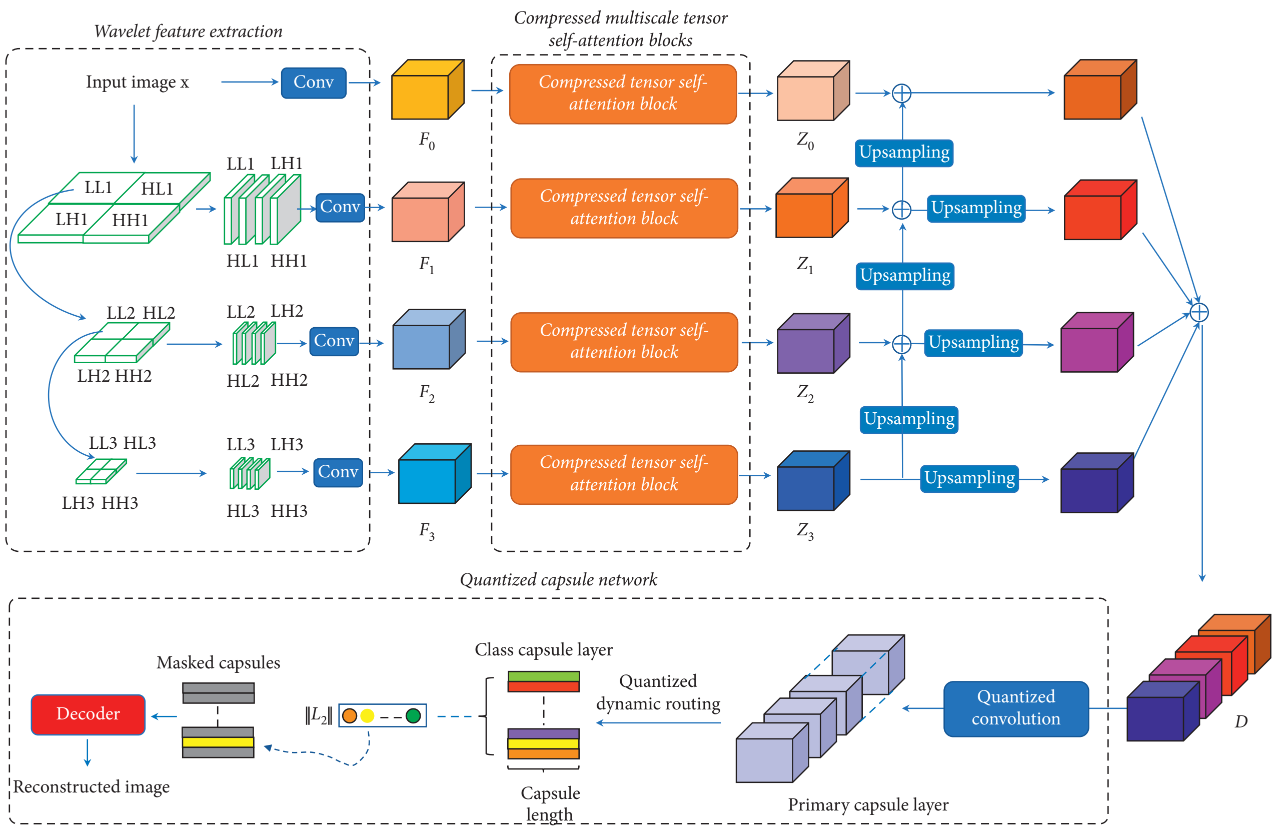


Figure 1: The architecture of CWTACapsNet. CWTACapsNet involves the wavelet feature extraction block, the compressed multiscale tensor self-attention block, and the quantized capsule network. ⊕ denotes tensor concatenation operator. The multilevel wavelet decomposition extracts the multiscale spectral features.

approximation solution which releases the resource burden of inference and speed up significantly.

These tensorized convolution feature maps are generated based on wavelet-extracted features. The input 3rd-order tensor can be viewed as a combination of its three mode-matricizations. Combining their outputs allows the *compressed tensor self-attention block* to make use of interchannel and intrachannel interdependencies. Moreover, the *Nyström-based self-attention module* involved in the *compressed tensor self-attention block* implements the self-attention computation to explore dependencies along corresponding mode in a more efficient way. The architectures of the *compressed tensor self-attention block* and the *Nyström-based self-attention module* are shown in Figure 2.

A mode-n-matricization of 3rd-order input tensor F_i, $i \in \{0, 1, 2, 3\}$, is the vector obtained by fixing all indices of F_i except for the nth dimension and can be seen as a generalization of matrix's rows and columns, $n \in \{1, 2, 3\}$. The mode-n-matricization of 3rd-order tensor$R^{I_1 \times I_2 \times I_3}$ is a case of matricization denoted as $X_i^{(n)}$ and arranges its mode-n-fibers to be the columns of the resulting matrix.

To simplify notations, we ignore the subscript. Let $X \in R^{h \times \ell}$ be the input matrix of the self-attention module, and it is projected using three matrices $W_V \in R^{\ell \times v}$, $W_K \in R^{\ell \times m}$, and $W_Q \in R^{\ell \times m}$ to extract feature representations $Q \in R^{h \times m}$, $K \in R^{h \times m}$, and $V \in R^{h \times v}$ as follows:

$$\begin{aligned} Q &= XW_Q, \\ K &= XW_K, \\ V &= XW_V. \end{aligned} \tag{4}$$

The output of the self-attention module is computed by

$$O = V + \alpha \text{softmax}\left(\frac{QK^T}{\sqrt{m}}\right)V, \tag{5}$$

where $\alpha > 0$ denotes the learnable coefficient and softmax$(\cdot)$ denotes a row-wise softmax normalization function. Then, generate tensor Y by reshaping O as tensor form.

As shown in equation (5), the self-attention mechanism requires calculating h^2 similarity scores between each pair of vectors, resulting in a complexity of $O(h^2)$ for both memory and time. Due to this quadratic dependence on the input length, the application of self-attention is limited to small size matrices (e.g., $h < 1000$) for edge devices. It is necessary to reduce the resource burden. Inspired by [37], we utilize Nyström technique to build a resource-efficient self-attention module. We rewrite the softmax operation in equation (5) as follows:

$$S = \text{softmax}\left(\frac{QK^T}{\sqrt{m}}\right) = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix}_{h \times h}, \tag{6}$$

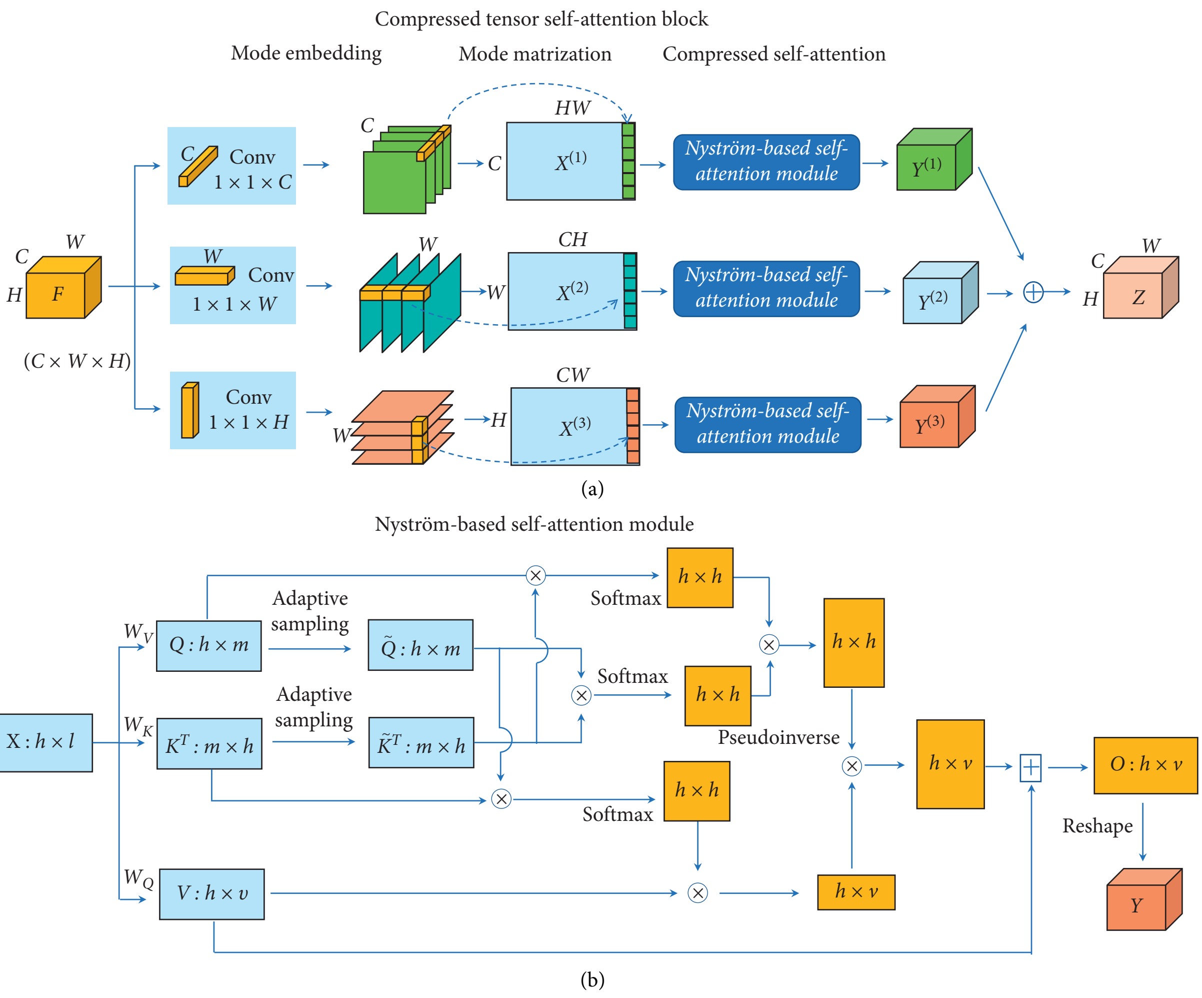

Compressed tensor self-attention block
Mode embedding
Mode matrization
Compressed self-attention
Conv 1 × 1 × C
Conv 1 × 1 × W
Conv 1 × 1 × H
(C × W × H)
Nyström-based self-attention module
(a)
Nyström-based self-attention module
Adaptive sampling
Softmax
Pseudoinverse
Reshape
(b)

Figure 2: (a) The architecture of compressed tensor self-attention block. The compressed tensor self-attention block involves the mode embedding via 1 × 1 convolution to realize mode matrization, and Nyström-based self-attention modules. (b) The architecture of Nyström-based self-attention module. Multidirectional interdependencies can be captured by the combination of both mode matrization and self-attention modules. ⊞ denotes the element-wise addition operator for matrices.

where $A_S \in R^{m\times m}, B_S \in R^{m\times (h-m)}, F_S \in R^{(h-m)\times m}$, and $C_S \in R^{(h-m)\times (h-m)}$. A_S denotes the selected matrix generated by sampling m columns and m rows from matrix S via some adaptive sampling strategy [38].

According to the Nyström method [37, 38], S can be approximated by

$$S \approx \widehat{S} = \begin{bmatrix} A_S & B_S \\ F_S & C_S \end{bmatrix} = \begin{bmatrix} A_S & B_S \\ F_S & F_S A_S^{\dagger} B_S \end{bmatrix}_{h\times h} = \begin{bmatrix} A_S \\ F_S \end{bmatrix} A_S^{\dagger} \begin{bmatrix} A_S & B_S \end{bmatrix}, \tag{7}$$

where $A_S^{\dagger}$ denotes the pseudoinverse (Moore–Penrose inverse) of A_S. C_S is approximated by $F_S A_S^{\dagger} B_S$.

The SVD of A_S can be written as $A_S = U_A \Sigma_A V_A^T$, where $U_A U_A^T = I$ and $V_A V_A^T = I$ denote orthogonal unitary matrices, and $\Sigma_A \in R^{m\times m}$ denotes the diagonal matrix whose diagonal elements are corresponding singular values of A_S. Then, pseudoinverse $A_S^{\dagger}$ can be computed by

$$A_S^{\dagger} = V_A \Sigma_A^{-1} U_A^T. \tag{8}$$

Submitting equations (8) into (7), we obtain

$$\widehat{S} = \begin{bmatrix} A_S \\ F_S \end{bmatrix} V_A \Sigma_A^{-1} U_A^T \begin{bmatrix} A_S & B_S \end{bmatrix}. \tag{9}$$

From equations (6)–(9), we can find that $\widehat{S}$ requires all entries in QK^T due to the softmax function, even though the approximation only needs to access a subset of the columns and rows of S, e.g., $\begin{bmatrix} A_S \\ F_S \end{bmatrix} \in R^{h\times m}$ corresponds to the first m columns of S (see equation (6)) and $\begin{bmatrix} A_S & B_S \end{bmatrix} \in R^{m\times h}$ corresponds to the first m rows of S. An efficient way is to approximate $\widehat{S}$ using subsampled matrices instead of whole one (i.e., the $h \times h$ matrix QK^T). Let $\widetilde{K}^T \in R^{m\times m}$ denote the matrix that consists of m columns of $K^T \in R^{m\times h}$, and $\widetilde{Q} \in R^{m\times m}$ denote the matrix that consists of m rows of $Q \in R^{h\times m}$. Then, we compute the approximations as follows:

$$\begin{bmatrix} A_S \\ F_S \end{bmatrix} \approx \text{softmax}\left(\frac{Q\widetilde{K}^T}{\sqrt{m}}\right),$$

$$[\, A_S \;\; B_S \,] \approx \text{softmax}\left(\frac{\widetilde{Q}K^T}{\sqrt{m}}\right), \tag{10}$$

$$A_S^{\dagger} \approx \left(\text{softmax}\left(\frac{\widetilde{Q}\widetilde{K}^T}{\sqrt{m}}\right)\right).$$

Based on equations (7)–(9), we can obtain the efficiently approximated $\widehat{S}$ as follows:

$$\widehat{S} \approx \text{softmax}\left(\frac{Q\widetilde{K}^T}{\sqrt{m}}\right)\text{softmax}\left(\frac{\widetilde{Q}\widetilde{K}^T}{\sqrt{m}}\right)\text{softmax}\left(\frac{\widetilde{Q}K^T}{\sqrt{m}}\right), \tag{11}$$

where $\widetilde{Q}$ and $\widetilde{K}^T$ are selected before the softmax operation, which means $\widehat{S}$ can be computed only using small submatrices instead of the whole one (the $h \times h$ matrix QK^T).

The output of each single compressed tensor self-attention module is computed by

$$O = V + \alpha\widehat{S}V. \tag{12}$$

Then, the output of the compressed tensor self-attention block (Figure 2(a)) can be generated by

$$Z = \Psi_1\left(O^{(1)}\right) \oplus \Psi_2\left(O^{(2)}\right) \oplus \Psi_3\left(O^{(3)}\right), \tag{13}$$

where Ψ_n denotes a reshape function which rearranges the matrix $O^{(n)}$ as the tensor of dimension $C \times H \times W$, $n \in \{1, 2, 3\}$, and $\oplus$ denotes the matrix concatenation operator.

2.3. Quantized Capsule Network. Aiming to overcome the deficiency and shortcoming of convolutional neural networks, a novel architecture of neural network called capsule networks was first introduced by Geoffrey Hinton [14]. A capsule is a set of neurons represented as a vector. The individual values are to capture features of an object, while the length of the vector shows the capsule activation probability. The first layer of capsules comes from the output of a convolution. This output is rearranged into vectors with a previously specified dimension (and is shrunk using the squashing function), which are used to compute the output of a next layer set of capsules. The algorithm with which the next layer capsules are computed using the current layer of capsules outputs is called dynamic routing. It takes predictions from the current layer capsules about the output of the next layer capsules and computes the actual output according to an agreement metric between predictions.

It should note that the superiority of capsule network leads to the heavy burden of computation and storage. To address this problem and make it easy to deploy on edge computing devices, we integrate share structure and quantization technique into capsule network and propose the quantized capsule network.

As shown in Figure 1, the input of quantized capsule network is generated by concatenating the output tensors of multiple compressed tensor self-attention blocks through some upsampling operation. From these concatenated tensors, the quantized convolutional layer extracts basic features. The primary capsule layer explores more detailed patterns from the extracted basic features:

$$\mathbf{u}_i = \text{Reshape}(\text{QConv}(Z)), \quad i = 1, \ldots, M_P, \tag{14}$$

where $\mathbf{u}_i \in R^p$ denotes the output of capsule i in the primary capsule layer, p denotes the dimension of primary layer capsule vector (or capsule vector length), M_P denotes the number of capsules in the primary capsule layer, Reshape$(\cdot)$ denotes the function that reshapes the output tensors into capsule vectors (the detailed description is provided in [14]), and QConv$(\cdot)$ denotes the quantized convolution operator (the detailed derivation of quantized convolution is provided in [39]).

Generally, the prediction vector generated by the primary layer capsule i, $\widehat{\mathbf{u}}_{j|i} \in R^c$, indicates how much the primary layer capsule i contributes to the class layer capsule j. $\widehat{\mathbf{u}}_{j|i}$ is given by

$$\widehat{\mathbf{u}}_{j|i} = \mathscr{W}_{ji}\mathbf{u}_i, \quad i = 1, \ldots, M_P, \; j = 1, \ldots, M_C, \tag{15}$$

where $\mathscr{W}_{ji} \in R^{c \times p}$ denotes the weight matrix between the primary layer capsule i and the class layer capsule j, c denotes the dimension of the class layer capsule vector, p denotes the dimension of primary layer capsule vector (or capsule vector length), and M_C and M_P denote the numbers of capsules in the class capsule layer and the primary capsule layer, respectively.

From equation (15), we can find that there are $M_C \times M_P$ weight matrices $\mathscr{W}_{ji}$, which leads to heavy computation and memory burden. To reduce the burden, we adopt two strategies. First, we utilize the shared structure of weight matrices (shown in Figure 3) as

$$\widehat{\mathbf{u}}_{j|i} = \widetilde{\mathscr{W}}_j\mathbf{u}_i, \quad i = 1, \ldots, M_P, j = 1, \ldots, M_C, \tag{16}$$

where $\widetilde{\mathscr{W}}_j \in R^{c \times p}$ denotes the transformation weight matrix corresponding to class layer capsule j (i.e., each class layer capsule shares its weight matrix to all primary layer capsules). Equation (16) indicates that the number of weight matrices is reduced from $M_C \times M_P$ to M_C.

Second, we propose the quantized dynamic routing process that implements the dynamic routing in a more efficient way (shown in Figure 4). For simplicity, we assume that p can be divided by $P(<p)$ with no reminder and $\widetilde{p} = (p/P)$. Let $\widetilde{\mathscr{W}}_j = [\widetilde{\omega}_j^{(1)}, \widetilde{\omega}_j^{(2)}, \ldots, \widetilde{\omega}_j^{(P)}] \in R^{c \times p}$ where $\widetilde{\omega}_j^{(\tau)} \in R^{c \times \widetilde{p}}$ denotes the τ-th submatrix of $\widetilde{\mathscr{W}}_j$, $\tau = 1, \ldots, P$, $j = 1, \ldots, M_C$. We train subcodebook for subspaces of weight matrices as follows:

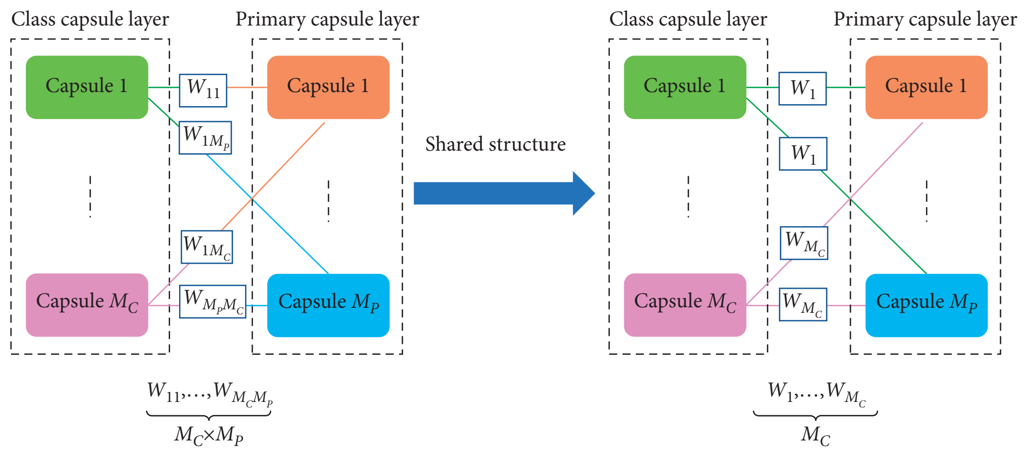


FIGURE 3: The shared structure of weight matrices of the capsule network.

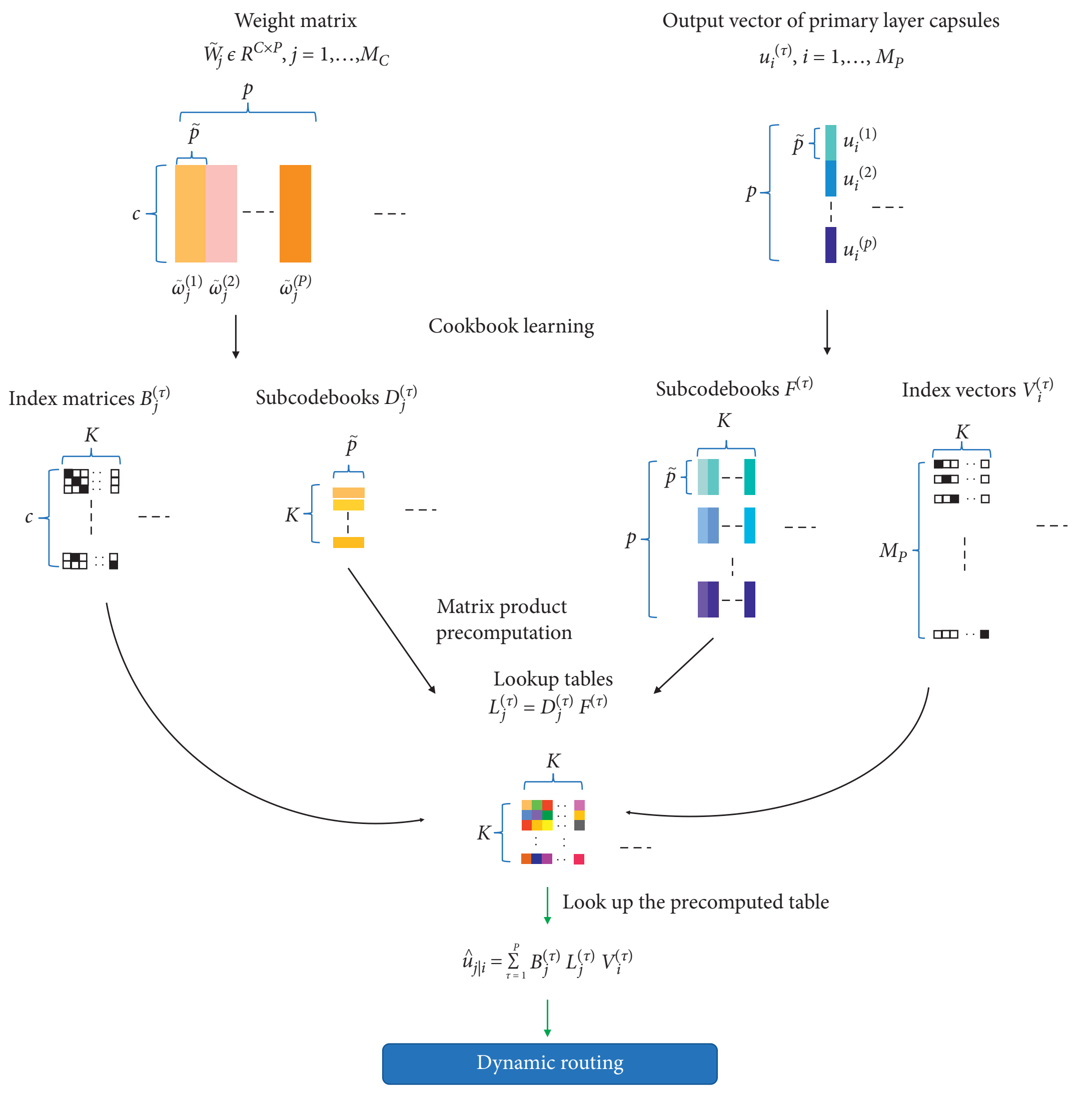

Weight matrix
$\tilde{W}_j \in R^{C\times P}, j = 1,\ldots,M_C$
Output vector of primary layer capsules
$u_i^{(\tau)}, i = 1,\ldots, M_P$
Cookbook learning
Index matrices $B_j^{(\tau)}$
Subcodebooks $D_j^{(\tau)}$
Subcodebooks $F^{(\tau)}$
Index vectors $V_i^{(\tau)}$
Matrix product precomputation
Lookup tables
$L_j^{(\tau)} = D_j^{(\tau)} F^{(\tau)}$
Look up the precomputed table
$\hat{u}_{j|i} = \sum_{\tau=1}^{P} B_j^{(\tau)} L_j^{(\tau)} V_i^{(\tau)}$
Dynamic routing

FIGURE 4: The workflow of quantized dynamic routing.

$$\min_{B_j^{(\tau)},\mathscr{D}_j^{(\tau)}} \sum_{j=1}^{M_C} \left\| B_j^{(\tau)} \mathscr{D}_j^{(\tau)} - \boldsymbol{\omega}_i^{(\tau)} \right\|_F^2 \quad (17)$$
$$s.t. B_j^{(\tau)} \in \{0,1\}^K, \mathscr{D}_j^{(\tau)} \in R^{K\times\widetilde{p}}, \tau = 1,\ldots,P,$$

where $\mathscr{D}_j^{(\tau)}$ denotes the subcodebook consists of K subcodewords for $\widetilde{\omega}_j^{(\tau)}$, $j = 1,\ldots,M_C$, and $B_j^{(\tau)}$ denotes the indexing matrix, and each row of $B_j^{(\tau)}$ only has one nonzero entry which specifies the quantization relationship between subvector and subcodeword. The alternative optimization algorithm, such as k-means clustering, is employed for learning $\mathscr{D}_j^{(\tau)}$ and $B_j^{(\tau)}$.

Let $\mathbf{u}_i^T = [((\mathbf{u}_i^{(1)})^T, (\mathbf{u}_i^{(2)})^T,\ldots,(\mathbf{u}_i^{(P)})^T)]$ where $\mathbf{u}_i^{(1)} \in R^{\widetilde{p}}$ denotes the τ-th subvector of $\mathbf{u}_i$, $\tau = 1,\ldots,P, i = 1,\ldots,M_P$. We train subcodebook for subspaces of primary layer capsule vectors as follows:

$$\min_{\mathbf{v}_j^{(\tau)},\mathscr{F}^{(\tau)}} \sum_{i=1}^{M_P} \left\| \mathscr{F}^{(\tau)} \mathbf{v}_i^{(\tau)} - \mathbf{u}_i^{(\tau)} \right\|_F^2 \quad (18)$$
$$s.t. \mathbf{v}_i^{(\tau)} \in \{0,1\}^K, \mathscr{F}^{(\tau)} \in R^{\widetilde{p}\times K}, \tau = 1,\ldots,P,$$

where $\mathscr{F}^{(\tau)}$denotes the subcodebook consists of K subcodewords for $\mathbf{u}_i^{(\tau)}$, $i = 1,\ldots,M_P$, and $\nu_i^{(\tau)}$denotes the index vector that only has one nonzero entry which specifies the quantization relationship between subvector and subcodeword. The alternative optimization algorithm, such as k-means clustering, can be employed for learning $\nu_i^{(\tau)}$ and$\mathscr{F}^{(\tau)}$.

Combining equations (17) and (18), we can rewrite equation (16) as

$$\widehat{\mathbf{u}}_{j|i} = \widetilde{\mathscr{W}}_j \mathbf{u}_i = \left[\widetilde{\boldsymbol{\omega}}_j^{(1)}, \widetilde{\boldsymbol{\omega}}_j^{(2)},\ldots,\widetilde{\boldsymbol{\omega}}_j^{(P)}\right] \begin{bmatrix} \mathbf{u}_i^{(1)} \\ \mathbf{u}_i^{(2)} \\ \vdots \\ \mathbf{u}_i^{(P)} \end{bmatrix} = \sum_{\tau=1}^{P} \widetilde{\boldsymbol{\omega}}_j^{(\tau)} \mathbf{u}_i^{(\tau)} \approx \sum_{\tau=1}^{P} B_j^{(\tau)} \mathscr{D}_j^{(\tau)} \mathscr{F}^{(\tau)} \mathbf{v}_i^{(\tau)}, \quad (19)$$

where$\widehat{\mathbf{u}}_{j|i} \in R^c$, $B_j^{(\tau)} \in \{0,1\}^K, \mathscr{D}_j^{(\tau)} \in R^{K\times\widetilde{p}}, \nu_i^{(\tau)} \in \{0,1\}^K, \mathscr{F}^{(\tau)} \in R^{\widetilde{p}\times K}, i = 1,\ldots,M_P, j = 1,\ldots, M_C$.

It is obvious that there are many replicate elements in the product of $\mathscr{D}_j^{(\tau)}\mathscr{F}^{(\tau)}$, after the parameter quantization. Therefore, it is unwise to compute the products in a one-by-one style. Instead, we first compute the results of the product $\mathscr{D}_j^{(\tau)}\mathscr{F}^{(\tau)}$, i.e., constructing the lookup table, as follows:

$$\mathscr{L}_j^{(\tau)} = \mathscr{D}_j^{(\tau)} \mathscr{F}^{(\tau)}, \quad (20)$$

where $\mathscr{L}_j^{(\tau)} \in R^{K\times K}$, $j = 1,\ldots,M_C, \tau = 1,\ldots,P$.

Then, in the application, we can look up the precomputed table instead of repeatedly computing which raises computational speed significantly. Hence, we can rewrite equation (19) as follows:

$$\widehat{\mathbf{u}}_{j|i} \approx \sum_{\tau=1}^{P} B_j^{(\tau)} \mathscr{D}_j^{(\tau)} \mathscr{F}^{(\tau)} \mathbf{v}_i^{(\tau)} = \sum_{\tau=1}^{P} B_j^{(\tau)} \mathscr{L}_j^{(\tau)} \mathbf{v}_i^{(\tau)}, \quad (21)$$

where the product $B_j^{(\tau)}\mathscr{L}_j^{(\tau)}\nu_i^{(\tau)}$ can be considered as the process of looking up the precomputed table $\mathscr{L}_j^{(\tau)}$ instead of the matrix multiplication operation.

According to the mechanism of capsule network, the input vector $\mathbf{z}_j$ of class layer capsule j can be computed by

$$\mathbf{z}_j = \sum_{i=1}^{M_P} \vartheta_{ij} \widehat{\mathbf{u}}_{j|i}, \quad j = 1,\ldots,M_C, \quad (22)$$

where ϑ_{ij} denotes the coupling coefficient determined by the iterative dynamic routing process (see Table 1). The routing part is actually a weighted sum of $\widehat{\mathbf{u}}_{j|i}$ with the coupling coefficient. The output vector of class layer capsule j is calculated by applying a nonlinear squashing function that can ensure short vectors to be shrunk to almost zero length, and long vectors get shrunk to a length slightly below one as

$$\boldsymbol{\rho}_j = \frac{\left\|\mathbf{z}_j\right\|^2}{1 + \left\|\mathbf{z}_j\right\|^2} \frac{\mathbf{z}_j}{\left\|\mathbf{z}_j\right\|^2}, \quad (23)$$

where ρ_j denotes the output vector of class layer capsule j.

Obviously, the capsule's activation function actually suppresses and redistributes vector lengths. Its output can be used as the probability of the entity represented by the current class capsule. The quantized dynamic routing algorithm is shown in Table 1.

We construct the whole loss function of the proposed CWTACaps by integrating the margin loss [14], reconstruction loss [14], and the quantization loss as follows:

$$\mathscr{L} = \mathscr{L}_{\max} + \lambda_{\text{re}}\mathscr{L}_{\text{re}} + \lambda_{\text{qu}}\mathscr{L}_{\text{qu}}, \quad (24)$$

where λ_{re} and λ_{qu} denote positive coefficients and $\mathscr{L}_{\max}$, $\mathscr{L}_{\text{re}}$, and $\mathscr{L}_{\text{qu}}$ denote the margin loss function, the reconstruction loss function, and the quantization loss function, respectively. They are defined by equations (25)–(27) as follows:

$$\mathscr{L}_{\max} = T_c \max\left(0, \varepsilon^+ - \boldsymbol{\rho}_j^2\right) + \eta_{\text{mar}}\left(1 - T_c\right)\max\left(\boldsymbol{\rho}_j - \varepsilon^-\right)^2, \quad (25)$$

$$\mathscr{L}_{\text{re}} = X - \widetilde{X}_F^2, \quad (26)$$

Table 1: Quantized dynamic routing algorithm.

```
Algorithm 1Quantized dynamic routing.
Input: $B_j^{(\tau)}, \mathscr{L}_j^{(\tau)}, v_i^{(\tau)}, t, l$
Output: $\rho_j$
(1) compute $\hat{\mathbf{u}}_{j|i}$ using the quantized version in equation (21)
(2) for all capsule $i$ in the primary layer and capsule $j$ in class layer: $b_{ij} \leftarrow 0$
(3) for $t$ iterations do
(4)     for all capsule $i$ in the primary layer and capsule $j$ in the class layer: $\vartheta_{ij} \leftarrow \mathrm{softmax}(b_{ij})$
(5)     for all capsule $j$ in the class layer: compute the input vector $\mathbf{z}_j$ using equation (22)
(6)     for all capsule $j$ in the class layer: compute the output vector $\rho_j$using equation (23)
(7)     for all capsule $i$ in the primary layer and capsule $j$ in the class layer: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}, \rho_j$
(8) Return $\rho_j$
```

$$\mathscr{L}_{\mathrm{qu}} = \sum_{\tau=1}^{P} \sum_{j=1}^{M_C} B_j^{(\tau)} \mathscr{D}_j^{(\tau)} - \widetilde{\boldsymbol{\omega}}_{Fj}^2 + \eta_{\mathrm{qu}} \sum_{\tau=1}^{P} \sum_{i=1}^{M_P} \mathscr{F}^{(\tau)} \mathbf{v}_i^{(\tau)} - \mathbf{u}_{iF}^{(\tau)2}, \tag{27}$$

where $T_c = 1$iif correct classification, $\varepsilon^+ = 0.9$ and $\varepsilon^- = 0.1$, η_{mar} and η_{qu} denote positive coefficients, usually selected as 0.5, and $\widetilde{X}$ denotes the reconstructed image.

3. Experiments

The aim of this section is to validate our proposed CWTA-CapsNet on three datasets: CUReT [40], DTD [41], and KTH-TIPS2-b [42]. For the CUReT dataset, we use the same subset as in [43], which contains 61 texture classes (92 images per class). The DTD dataset contains 47 classes (120 images per class). KTH-TIPS2-b contains 11 classes. Each class in KTH-TIPS2-b contains 432 images which are resized to 256×256 pixels. Besides CWTACapsNet, five state-of-the-art methods, T-CNN [13], FV-CNN [8], SI-LCvMSP [1], Wavelet CNNs [11], and CapsNet [44], are employed for performance comparison.

Models in experiments are trained under Ubuntu 16.04 with i7-8700 CPU, 64G RAM, and GeForce GTX Titan-XP GPU, and our proposed CWTACapsNet is deployed on Jetson TX2. To provide a direct comparison with published results, parameters of five state-of-the-art methods are set according to previous studies [1, 8, 11, 13, 44]. We use an exponential decay learning policy, with an initial learning rate of 0.001, 2000 decay steps, and 0.96 decay rate. We employ Adam optimizer to adjust the weights of CWTA-CapsNet in the training process. The batch size is set as 32. We implement data augmentation through rotating images with a random angle between 0° and 90°. We use 3 routing iterations to update capsule parameters in CWTACapsNet. The number of wavelet level in CWTACapsNet is selected according to the tradeoff between validation accuracy and network parameter amount. We thus choose 3-level wavelet decomposition. The learnable coefficient α is selected as 0.1, and λ_{re} and λ_{qu} are selected as 0.001 and 0.0013, respectively.

Table 2 illustrates classification accuracies and standard derivations of six methods. Table 2 indicates that CWTA-CapsNet achieves the best performance and is more stable than other methods. The tensor attention block makes CWTACapsNet be able to capture multidirectional dependencies while other methods cannot. FV-CNN performs better than CapsNets. FV-CNN and CapsNets both deal with pooling operation, and FV-CNN has some specific design to capture texture information. CNN-based texture classification methods tend to be limited by the lack of diversity of convolution filters. The multilevel wavelet decomposition extends both spatial and frequency features, which raises diversity of convolution filters and improves performance.

Table 2: Classification accuracies (%) of six texture classification methods on three texture datasets.

Method	Datasets		
	CUReT	DTD	KTH-TIPS2-b
Wavelet CNNs	78.95 ± 0.85	59.8 ± 0.92	74.2 ± 1.21
T-CNN	99.5 ± 0.4	55.8 ± 0.8	73.2 ± 2.2
CapsNets	92.4 ± 0.98	70.98 ± 1.03	73.83 ± 1.12
FV-CNN	95.7 ± 1.08	75.5 ± 0.8	81.5 ± 2.0
SI-LCvMSP	96.44 ± 0.75	76.7 ± 0.78	96.1 ± 1.02
CWTACapsNet	99.7 ± 0.22	81.52 ± 0.47	97.15 ± 0.55

Table 3: Classification accuracies (%) of six texture classification methods on three noisy texture datasets.

Method	Datasets		
	CUReT	DTD	KTH-TIPS2-b
Wavelet CNNs	73.35 ± 1.35	56.18 ± 1.42	70.4 ± 1.65
T-CNN	67.64 ± 1.56	50.62 ± 1.44	68.8 ± 3.72
CapsNets	89.32 ± 1.08	67.08 ± 1.29	69.1 ± 1.48
FV-CNN	92.7 ± 1.54	71.5 ± 1.8	76.5 ± 3.52
SI-LCvMSP	89.6 ± 1.45	71.7 ± 1.38	91.58 ± 1.72
CWTACapsNet	99.1 ± 0.33	79.82 ± 0.54	96.9 ± 0.39

We add 10% white noise into texture datasets to evaluate robustness. Table 3 shows the performance of noisy datasets. Figure 5 shows accuracy for pure and noisy data. Figure 6 shows accuracy standard derivations (std) for pure and noisy data.

From Table 3 and Figures 5 and 6, we can find that CWTACapsNet achieves the best accuracy and robustness. Although CapsNets and CWTACapsNet are both based on capsule layer, CWTACapsNet significantly outperforms CapsNets. The memory requirement of CapsNets in the experiments is about 272M, while our proposed CWTACapsNet only requires 23.2 M with about $10\times$ speed-up. CWTACapsNet can be deployed and run on Jetson TX2, while CapsNets requires too much resource that Jetson TX2 hardly supported. The superiority of CWTACapsNet relies on three factors: the

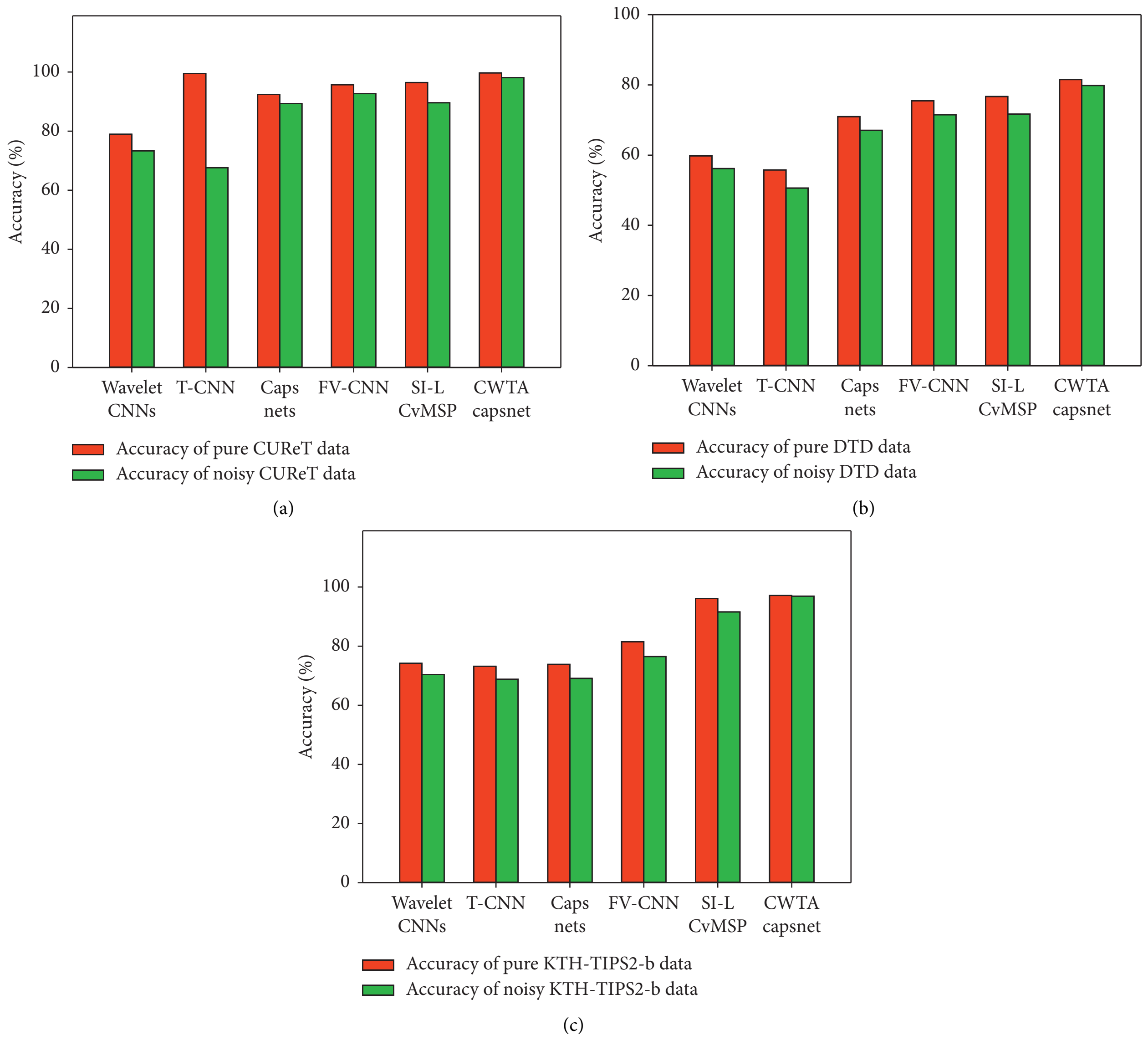
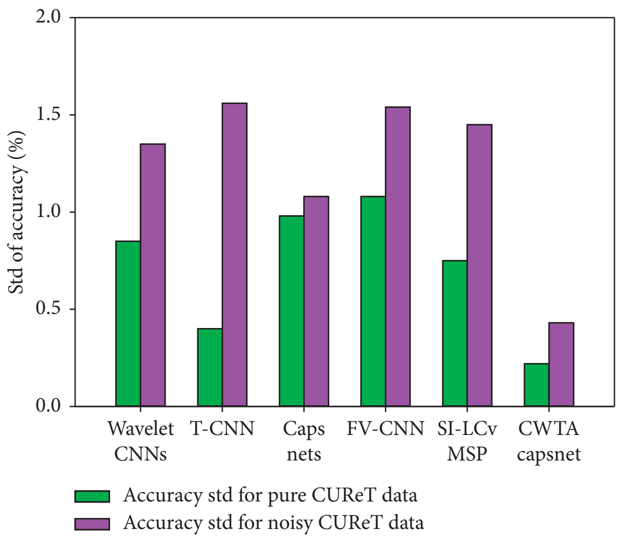
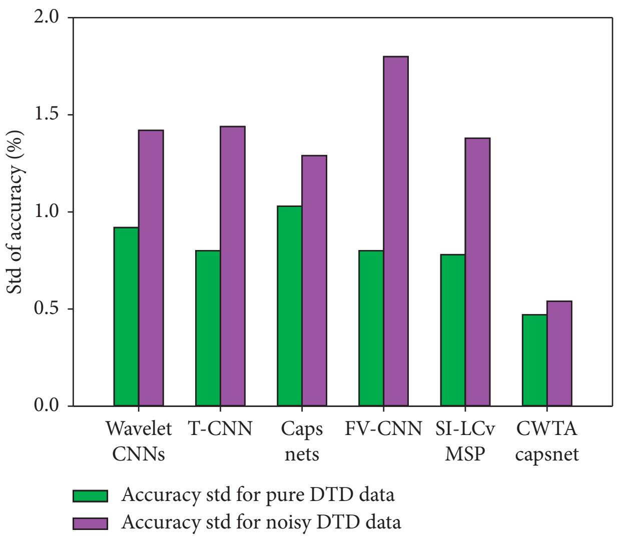


FIGURE 5: The accuracy of six texture classification methods for pure and noisy texture datasets, i.e., (a) CUReT, (b) DTD, and (c) KTH-TIPS2-b.

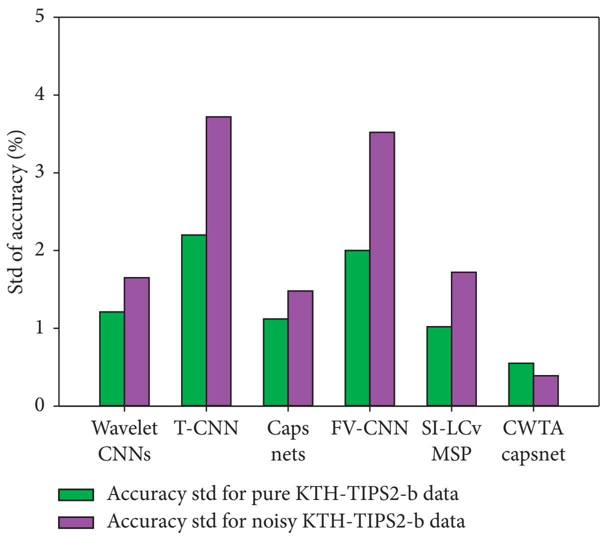
multilevel wavelet decomposition extends features from spatial space to frequency space, the tensor attention block explores relationships from all possible directions and captures the dependencies cross channels, and the quantized dynamic routing significantly reduces memory requirement. Experimental results validate the effectiveness of CWTACapsNet.



(a)



(b)



(c)

FIGURE 6: The accuracy of six texture classification methods for pure and noisy texture datasets, i.e., (a) CUReT, (b) DTD, and (c) KTH-TIPS2-b.

4. Conclusion

In order to make capsule network efficiently explore spatial and spectral features and capture multidirectional channel dependencies, this paper proposes a novel capsule network named compressed wavelet tensor attention capsule network (CWTACapsNet). In CWTACapsNet, the compressed multiscale wavelet transform is designed to extract multiscale spectral features in frequency domain; the tensor attention blocks utilize matrization to capture multiple directional dependencies across convolutional channels in terms of each scale information; furthermore, we propose quantized dynamic routing process for speeding up and storage reduction. Experimental studies have shown that the proposed CWTACapsNet provides the best performance on both classification result and antinoise robustness; moreover, CWTACapsNet significantly reduces the computational and storage complexities. In the future, we will incorporate parallel computation methods into CWTACapsNet to further improve efficiency.

Data Availability

The authors approve that data used to support the finding of this study are publicly available. The datasets can be achieved from the links provided by [40–42]. CUReT Dataset is available at https://www.cs.columbia.edu/CAVE/software/curet/html/download.h

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Chinese National Natural Science Foundations (nos. 61972104 and 61571141).

References

[1] N. Alpaslan and K. Hanbay, "Multi-scale shape index-based local binary patterns for texture classification," *IEEE Signal Processing Letters*, vol. 27, pp. 660–664, 2020.

[2] Q. Kou, D. Cheng, H. Zhuang, and R. Gao, "Cross-complementary local binary pattern for robust texture classification," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 129–133, 2019.

[3] S. S. Moghadasian and S. Gazor, "Sparsely localized time-frequency energy distributions for multi-component LFM signals," *IEEE Signal Processing Letters*, vol. 27, pp. 6–10, 2020.

[4] X. Dong, H. Zhou, and J. Dong, "Texture classification using pair-wise difference pooling-based bilinear convolutional neural networks," *IEEE Transactions on Image Processing*, vol. 29, pp. 8776–8790, 2020.

[5] L. Liu, J. Chen, P. Fieguth, G. Zhao, R. Chellappa, and M. Pietikäinen, "From BoW to CNN: two decades of texture representation for texture classification," *International Journal of Computer Vision*, vol. 127, no. 1, pp. 74–109, 2019.

[6] A. Humeau-Heurtier, "Texture feature extraction methods: a survey," *IEEE Access*, vol. 7, pp. 8975–9000, 2019.

[7] X. Dong, J. Dong, and M. Chantler, "Perceptual texture similarity estimation: an evaluation of computational features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, 2020.

[8] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3828–3836, Boston, MA, USA, June 2015.

[9] X. Dong and M. J. Chantler, "Perceptually motivated image features using contours," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5050–5062, 2016.

[10] Y. Kim, B. Ham, M. N. Do, and K. Sohn, "Structure-texture image decomposition using deep variational priors," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2692–2704, 2019.

[11] S. Fujieda, K. Takayama, and T. Hachisuka, "Wavelet convolutional neural networks for texture classification," 2017, http://arxiv.org/abs/1707.07394.

[12] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *International Journal of Computer Vision*, vol. 118, no. 1, pp. 65–94, 2016.

[13] V. Andrearczyk and P. F. Whelan, "Using filter banks in convolutional neural networks for texture classification," *Pattern Recognition Letters*, vol. 84, pp. 63–69, 2016.

[14] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proceedings of the 2017 The Thirty-First Annual Conference on Neural Information Processing Systems*, pp. 3856–3866, Long Beach, CA, USA, May 2017.

[15] D. Xu, Z. Tian, R. Lai, X. Kong, Z. Tan, and W. Shi, "Deep learning based emotion analysis of microblog texts," *Information Fusion*, vol. 64, pp. 1–11, 2020.

[16] C. Xiang, L. Zhang, Y. Tang, W. Zou, and C. Xu, "MS-capsnet: a novel multi-scale capsule network," *IEEE Signal Processing Letters*, vol. 25, no. 12, pp. 1850–1854, 2018.

[17] S. Zhang, W. Zhao, X. Wu, and Q. Zhou, "Fast dynamic routing based on weighted kernel density estimation," *Concurrency and Computation: Practice and Experience*, vol. 5281, pp. 1–10, 2019.

[18] D. Guo, H. Wang, S. Wang, and M. Wang, "Textual-visual reference-aware attention network for visual dialog," *IEEE Transactions on Image Processing*, vol. 29, pp. 6655–6666, 2020.

[19] T. Xi, "Design of English diagnostic practice sentence repetition recognition system based on matching tree and edge computing," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6651145, 8 pages, 2021.

[20] Z. Xie, L. Hu, Y. Huang, and J. Pang, "A semiopportunistic task allocation framework for mobile crowdsensing with deep learning," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6643229, 15 pages, 2021.

[21] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2021.

[22] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[23] M. Shafiq, Z. Tian, A. K. Bashir, A. Jolfaei, and X. Yu, "Data mining and machine learning methods for sustainable smart cities traffic classification: a survey," *Sustainable Cities and Society*, vol. 60, Article ID 102177, 2020.

[24] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "IoT malicious traffic identification using wrapper-based feature selection mechanisms," *Computers & Security*, vol. 94, Article ID 101863, 2020.

[25] M. Shafiq, Z. Tian, A. K. Bashir, K. Cengiz, and A. Tahir, "SoftSystem: smart edge computing device selection method for IoT based on soft set technique," *Wireless Communications and Mobile Computing*, vol. 2020, Article ID 8864301, , 2020.

[26] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020.

[27] M. Li, Y. Sun, H. Lu, S. Maharjan, and Z. Tian, "Deep reinforcement learning for partially observable data poisoning attack in crowdsensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6266–6278, 2020.

[28] S. Su, Z. Tian, S. Liang, S. Li, S. Du, and N. Guizani, "A reputation management scheme for efficient malicious vehicle identification over 5G networks," *IEEE Wireless Communications*, vol. 27, no. 3, pp. 46–52, 2020.

[29] Z. Gu, L. Wang, X. Chen et al., "Epidemic risk assessment by A novel communication station based method," *IEEE Transactions on Network Science and Engineering*, p. 1, 2021.

[30] Y. Wang, Z. Tian, Y. Sun, X. Du, and N. Guizani, "LocJury: an IBN-based location privacy preserving scheme for IoCV," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2020.

[31] Y. Wang, Z. Tian, H. Zhang, S. Su, and W. Shi, "A privacy preserving scheme for nearest neighbor query," *Sensors (Basel, Switzerland)*, vol. 18, no. 8, p. 2440, 2018.

[32] W. Jiang, Z. Tian, H. Zhang, and X. Song, "A stochastic game theoretic approach to attack prediction and optimal active defense strategy decision," in *Proceedings of the 2008 IEEE International Conference on Networking, Sensing and Control*, pp. 648–653, Sanya, China, April 2008.

[33] W. Jiang, B. Fang, H. Zhang, Z. Tian, and X. Song, "Optimal network security strengthening using attack-defense game model," in *Proceedings of the 2009 Sixth International*

Conference on Information Technology: New Generations, pp. 475–480, Las Vegas, NV, USA, April 2009.

[34] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[35] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, "Multi-level wavelet-CNN for image restoration," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 886–88609, Salt Lake City, UT, USA, 2018.

[36] F. Babiloni, I. Marras, G. Slabaugh, and S. Zafeiriou, "TESA: tensor element self-attention via matricization," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13942–13951, Seattle, WA, USA, June 2020.

[37] Y. Xiong, Z. Zeng, R. Chakraborty et al., "Nyströmformer: a nyström-based algorithm for approximating self-attention," 2021, http://arxiv.org/abs/2102.03902.

[38] S. Wang, A. Gittens, and M. W. Mahoney, "Scalable kernel K-means clustering with Nyström approximation: relative-error bounds," *Journal of Machine Learning Research*, vol. 20, no. 12, pp. 1–49, 2019.

[39] J. Cheng, J. Wu, C. Leng, Y. Wang, and Q. Hu, "Quantized CNN: a unified approach to accelerate and compress convolutional networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4730–4743, 2018.

[40] M. Varma and A. Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 11, pp. 2032–2047, 2009.

[41] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, Columbus, OH, USA, June 2014.

[42] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh, "On the significance of real-world conditions for material classification," in *Proceedings of the 2004 8th European Conference on Computer Vision*, pp. 253–266, Prague, Czech Republic, May 2004.

[43] L. Liu and P. W. Fieguth, "Texture classification from random features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 574–586, 2012.

[44] B. Mamidibathula, S. Amirneni, S. S. Sistla, and N. Patnam, "Texture classification using capsule networks pattern recognition and image analysis," *Pattern Recognition and Image Analysis*, vol. 11867, pp. 589–599, 2019.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5587511, 10 pages
https://doi.org/10.1155/2021/5587511

Research Article

Employing Deep Learning and Time Series Analysis to Tackle the Accuracy and Robustness of the Forecasting Problem

Haseeb Tariq,[1] Muhammad Kashif Hanif,[1] Muhammad Umer Sarwar,[1] Sabeen Bari,[2] Muhammad Shahzad Sarfraz,[3] and Rozita Jamili Oskouei[4]

[1]*Department of Computer Science, Government College University, Faisalabad, Pakistan*
[2]*Griffith College Dublin, Dublin, Ireland*
[3]*Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad Chiniot-Faisalabad Campus, Chiniot 35400, Pakistan*
[4]*Department of Computer Science and Information Technology, Mahdishahr Branch, Islamic Azad University, Mahdishahr, Iran*

Correspondence should be addressed to Rozita Jamili Oskouei; rozita2020j@gmail.com

Received 3 February 2021; Revised 15 February 2021; Accepted 11 March 2021; Published 1 April 2021

Academic Editor: Shah Nazir

Crime is a bone of contention that can create a societal disturbance. Crime forecasting using time series is an efficient statistical tool for predicting rates of crime in many countries around the world. Crime data can be useful to determine the efficacy of crime prevention steps and the safety of cities and societies. However, it is a difficult task to predict the crime accurately because the number of crimes is increasing day by day. The objective of this study is to apply time series to predict the crime rate to facilitate practical crime prevention solutions. Machine learning can play an important role to better understand and analyze the future trend of violations. Different time-series forecasting models have been used to predict the crime. These forecasting models are trained to predict future violent crimes. The proposed approach outperforms other forecasting techniques for daily and monthly forecast.

1. Introduction

Urbanization is becoming a global trend [1]. As the city grows, different management challenges increase on a daily basis. Nowadays, crime is a problematic social matter. The crime rate in big cities is higher than in smaller localities. One of the main problems in many countries is the increase in the crime rate in urban areas. With the increasing amount of crimes, crime evaluation methods are needed to reduce the crime [2]. The criminal activities can be reduced by a distribution of patrol officers according to the crime rate. However, it is hard to predict future crimes accurately and efficiently.

Crimes can be categorized into different types such as violent and nonviolent crimes. A violent crime is a crime in which criminals threaten a targeted person. These crimes are considered more serious than nonviolent crimes [3]. A violent act is composed of different offenses such as homicide, aggravated assault, battery, kidnapping, robbery, murder, and forcible rape [4, 5]. Violent crime may or may not happen with the weapon. Different countries also have distinct methods of recording and crime reporting.

The main challenge is to analyze the increasing volume of criminal data correctly and efficiently [6]. Mostly, security forces lack the tools and skills to recognize effective patterns in these enormous data. Data mining methods can be used to extract valuable information to enhance the efficiency of the city police and enable the officers to make better use of the confined resources. In addition, advanced analytic methods can be integrated with current planning tools. This can enable crime investigators to access huge databases without the need for training from data scientists.

Forecasting is used to project past and present events into the future. Forecasting techniques identify, model, and extrapolate the patterns found in historical data. Forecasting problems can be categorized into short, medium, and long term based on prediction periods. The majority of forecasting problems use the time series data. A time series is a time-oriented sequence of observations. Time series analysis produces models that can help to understand the underlying causes by using the observed time series. Time series models use the statistical properties of the historical data to predict future patterns and trends [7].

Conventional time-series analysis models such as autoregressive integrated moving-average (ARIMA) [8] and machine learning models such as artificial neural network (ANN) are insufficient to tackle the forecasting problem of criminal data. Researchers have also employed the hybrid models to denoise the data and to find the linear and nonlinear patterns in the data to improve the performance of the forecasting [9, 10].

The objective of the current study is to evaluate the predictive capacity of the models for a short- and medium-term forecast for criminal data. This will help in optimal decision-making and resource management. This work compares different time-series analysis models and machine learning models, i.e., ARIMA, simple exponential smoothing (SES), Holt–Winters exponential smoothing (HW), and recurrent neural network (RNN), to predict the crime trends.

The rest of the paper is organized into different sections. Section 2 discusses different time series forecasting used in this work. Section 3 presents the related work. Section 4 describes the time-series forecasting methodology. Section 5 presents the experimental evaluation of the proposed technique. The outcomes are concluded in Section 6.

2. Time Series Forecasting

Structured series of data points listed at an equal-spaced time is called time series. Time series analysis can be separated into two parts. The first part is to obtain the structured underlying pattern of the ordered data. The second part narrates to fit a model for future prediction. The most challenging part that involves mathematical calculations is the fitting part of the time series. Time series can be used for univariate and multivariate analyses [11]. This section discusses different time-series forecasting models to predict future crimes.

2.1. ARIMA Model. ARIMA model is a widely used time-series forecasting model introduced by Box and Jenkins in 1970 [12]. ARIMA model is a general linear stochastic model which is the combination of autoregressive and moving-average models [13–15]. An autoregressive model uses a linear combination of past values to predict the variable of interest. The moving-average model uses the past predictions' errors similar to the regression model [16]. It carries a limited number of parameters such as (p, d, q), where p represents the order of the AR model, d is the degree of differencing, and q is the moving-average model [17, 18].

$$y_t = c + \varphi_1 y_{t-1} + \cdots + \varphi_p y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t, \tag{1}$$

where $\varphi_1, \ldots, \varphi_p$ are the parameters for the autoregressive model, $\theta_1, \ldots, \theta_q$ are the parameters for the moving-average model, $y_{t-1}, \ldots, y_{t-p}$ are defined as the past values (lags), e_t is the white noise, and y_t is the difference at degree d of the original series of time series.

2.2. Exponential Smoothing Methods. This time series forecasting is used for univariate data. The exponential smoothing technique processes smoothing parameters determined from past data. For prediction, new observations can have greater value than the previous observations [19]. Smoothing variables are determined by minimizing the mean absolute percentage error (MAPE) and root mean square error (RMSE).

2.2.1. Simple Exponential Smoothing Method. Simple exponential smoothing is the simplest method that is suitable for stationary series. It is a time-series forecasting approach for a single parameter without a trend and seasonality. SES models are generally based on the assumption that time series should be oscillating at a constant level or slowly changing over time [20]. This method requires little computation. Let $z_1, z_2, z_3, \ldots, z_n$ be a time series. Formally, SES can be computed as

$$\hat{z}_{t+1} = \alpha z_t + (1 - \alpha)\hat{z}_t, \tag{2}$$

where z_t is the actual known series of time period i, $\hat{z}_t$ is the forecast value of γ for time period i, $\hat{z}_{t+1}$ is the forecast value for time period $i + 1$, and α is the smoothing constant. The prediction $\hat{z}_{t+1}$ is based on the weighted nearest observation z_t, its weight α, the weight of the nearest prediction $\hat{z}_t$, and the weight $1 - \alpha$ [21].

2.2.2. Holt–Winters Exponential Smoothing Method. Holt–Winters exponential smoothing method was designed in 1960 by extending the exponential smoothing method. HW is applied when data are in the stationary form. For the calculation of the prediction measures, all the data values need to be in series. This method is suitable when data are with the trend and seasonality [22]. The basic equations applied in each update cycle for level L, trend b, seasonality S, and forecast F at time t are

$$\begin{aligned} L_t &= \alpha\left(x_t - S_{t-c}\right) + (1 - \alpha)\left(d_{t-1} + b_{t-1}\right), \\ b_t &= \beta\left(d_t - d_{t-1}\right) + (1 - \beta)b_{t-1}, \\ S_t &= \gamma\left(x_t - d_t\right) + (1 - \gamma)S_{t-s}, \\ F_{t+m} &= d_t + b_t m + S_{t-c+m}, \end{aligned} \tag{3}$$

where $t = 1, 2, 3, \ldots$. L_t and b_t estimate the level of the series and the slope of the series at time t, respectively.

Exponential smoothing is not suitable for seasonal data including trends or cycles. However, the HW model uses a modified form of exponential smoothing. It applies three exponential smoothing formulae called triple exponential smoothing. First, the average is computed to give locals the average of the series. Second, the trend is smooth, and finally, smooth each subseries seasonal estimates for each season separately. The exponential smoothing formula applies to a series of trend and constant seasonal elements using HW addition and multiplication methods. An additive method is applied when the season changes through the series are roughly unchanged. The multiplicative method is employed when changes are in proportional series [22]. This study is only applicable to the HW additive model.

2.3. Recurrent Neural Network. RNN is a type of ANN which has input, hidden, and output units. Generally, the RNN model has a unidirectional flow of information from input layers to hidden layers. It remembers end-to-end working of the model [23]. Figure 1 explains the RNN framework for modeling time-series observations. A directional loop can help to remember when to make a decision, what is an input of the current node, and what it had learned from the inputs received previously. Using the previous sequence samples may help understand the current sample. RNN can work well on time series because of its capability of remembering the previous input received using the internal memory. This can help to make the RNN forecast accurately.

Long short-term memory (LSTM) networks are modified versions of the RNN that can help to solve the short- and long-term dependencies which make it easier to remember previous data. LSTM networks are trained using back-propagation through time which helps to overcome the vanishing gradient problem. Traditional neural networks have neurons, while LSTM networks have memory blocks connected through sequential layers. Each module contains gates that can handle module status and outputs. The gated formation of the LSTM network manages its memory state. The use of neural networks reduces the need for extensive feature engineering and allows training of large datasets [25].

The difference between LSTM and RNN is an internal unit state which is also transmitted along with the hidden state. The LSTM block receives the input sequence and then uses a gate activation unit to decide if it is dynamic. This action creates a state change and adds information that conditionally passes through the block. Gates make blocks much better than the classic neurons and enable them to memorize current streams.

The weight of the gates can be learned during the training phase. The gating function controls the input, remembers the content in the internal state variables, and handles the output that makes the LSTM unit flexible. In LSTM cells, there are three types of gates, i.e., input, forget, and output (Figure 2). Each unit of LSTM has a cell which has a state c_t at time t. The cell read/modify action is controlled using the input gate i_t, forget gate f_t, and output gate o_t. At each time step, the LSTM unit receives the input from two external sources at each of the four terminals, i.e., the three gates and the input [26].

3. Related Work

This section discusses the popular existing techniques to predict crimes. However, these techniques can have constraints. Specific algorithms can be chosen at the identification, feature, and modeling stages. These algorithms can identify and depict natural trends, models, and data relationships.

In recent years, ML algorithms have become increasingly popular and can be used for prediction. Researchers have analyzed the working of criminal activities by using these models in time series such as ARIMA, SES, HW, and RNN models by considering these accuracy metrics. Different researchers have worked on identifications of violations in different states of the United States by examining different datasets. Information such as the trend and seasonality of the crime was extracted to help people and peace enforcement agencies. The crime databases depend on places to identify violation hotspots. There exist a number of online map applications that can show the correct place of the crime and type of offense in any part of the city. Criminal sites can be identified precisely [27]. On the contrary, the historical data and present approaches primarily determine the criminal act [27]. Predictive police are working in Philadelphia where law enforcement agencies highlight and forecast crimes based on locations [28].

Marzan et al. [29] evaluated daily and weekly crime patterns using linear regression, multilayer perceptron, Gaussian processes, and sequential minimal optimization regression. They forecasted the outcome for 10 days and 10 weeks. History is the primary basis of crime forecasting. Cesario et al. [6] used autoregressive models to analyze and forecast crimes in selected regions of Chicago. They examined a number of crimes and violations over time and separated them into trends, seasonality, and random signals. They predicted the crime for one and two years. The downside is an analysis only for a specific area, and it is intended for length prediction.

Moreover, researchers have analyzed the effectiveness and accuracy of algorithms for crime predictions and other potential applications for peace enforcement analysis such as identifying real crime locations, crime profiles, and discovering criminal trends. The most important component is the accuracy of creating new information (based on previous observations), which can reduce the crime rate. Borowik et al. [30] applied prophet forecasting and spectral analysis for real time series in Poland. The authors determined the weekly and annual seasonal patterns for long-period trends in selected sorts of events. There is still a considerable change in crime that cannot be taken by the expected model. It has commonly been assumed that anticipated levels are beneficial for more appropriate allocation of peace enforcement agencies [30].

Chen et al. [17] applied the ARIMA model for short-term forecasting on property crimes. They compared the forecasting results with simple exponential smoothing and Holt two-parameter exponential smoothing model. By the given data for 50 weeks of property crime, they forecasted one week ahead from the given observations using the ARIMA model [17]. However, they only compared straightforward techniques and measured the amount of crime over the whole city and not over districts or grid cells. Their approach used grid cells. The data also lacked historical information.

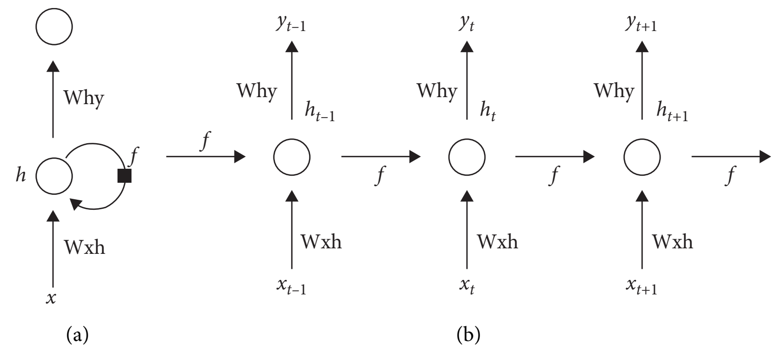


FIGURE 1: Example of (a) folded and (b) unfolded RNN [24].

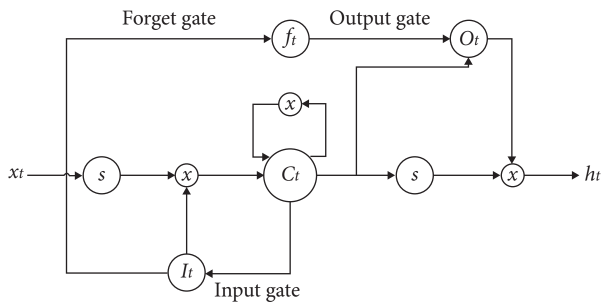


FIGURE 2: LSTM unit and its components [26].

Feng et al. [31] investigated crimes in Chicago, Philadelphia, and San Francisco by applying the Holt–Winters model. Firstly, the authors predicted the trend of crime in the next few years. After that, the category of crime was forecasted for time and location. For this, they collected multiple classes in a larger set and made an attribute selection. The outcomes showed that the tree classification models had performed better on classification tasks when compared with naive Bayesian methods and KNN. Holt–Winters model multiplicative seasonality provided good results when predicting the criminal tendency [31].

Singh [32] described a method to predict the crime for one week by taking 30 days' input of data using LSTM. He compared the performance of different models. Gated recurrent units have good crime prediction performance when compared with the traditional ARIMA model, artificial neural network, convolutional neural network, and RNN with its type [32]. Catlett et al. [33] proposed a predictive approach based on autoregressive and spatial analysis models to detect high-risk crime regions and forecasted crime trends.

Existing techniques have used different algorithms and strategies to forecast different types of data. Some techniques have been used for stationary data, while others are used for univariate data. Moreover, a majority of existing techniques are used to forecast a specific crime and focus on short-term prediction. In this study, the data are made stationary, and autocorrelation is used to find the correlation of lagged values. The proposed technique applies RNN along with LSTM to avoid the exploding gradient and vanishing gradient problems.

4. Methodology

This section describes the methods for data collection, preparation of the dataset, model testing, and training. The dataset is collected from the official website of Philadelphia crime [34] through the API. The dataset contains information on different kinds of violent crime from 2006 to 2016. The crime time and location information are used to forecast short- and medium-term crimes.

Figure 3 depicts the methodology used in this research for the violent crime dataset. First, data preprocessing is applied to transform raw data into clean data.

Data preprocessing includes removing unnecessary attributes, filling empty cells, and adding multiple related features. Data cleaning is employed to remove erroneous values. This is the most important and challenging part to achieve high accuracy. The features which contain more than 60% missing values are dropped since they are not helpful for further analysis. Moreover, outliers and duplicate values are filtered out. In the next step, data are standardized and normalized for further analysis.

Dimensionality reduction techniques reduce the high-dimensional data to low-dimensional data. In this study, we have applied the principal component analysis (PCA) method which provides linear mapping based on an eigenvector search. PCA provides different approaches to reduce the feature space dimensionality [35, 36]. In this study, the dataset is split into 70 : 30 ratio, i.e., 70% of the data is used for training, while 30% is used for the testing purpose.

The most important step in the workflow is choosing an appropriate model. Time series algorithms are used to predict the number of offenses that may occur in the next few years. Time series forecasting can be applied to time-dependent values. In this work, classical statistical methods are used along with machine learning techniques. Next, data exploration techniques are applied to understand the hidden insights of the dataset. Visualizations of a dataset are performed to find the trends and seasonality patterns in the data without transforming or changing the dataset.

Figure 4 illustrates the raw data visualization in the order of total crime occurrences from the observed data on a daily, monthly, and yearly basis, respectively. Crime data are plotted as a time series along the *X*-axis and the number of

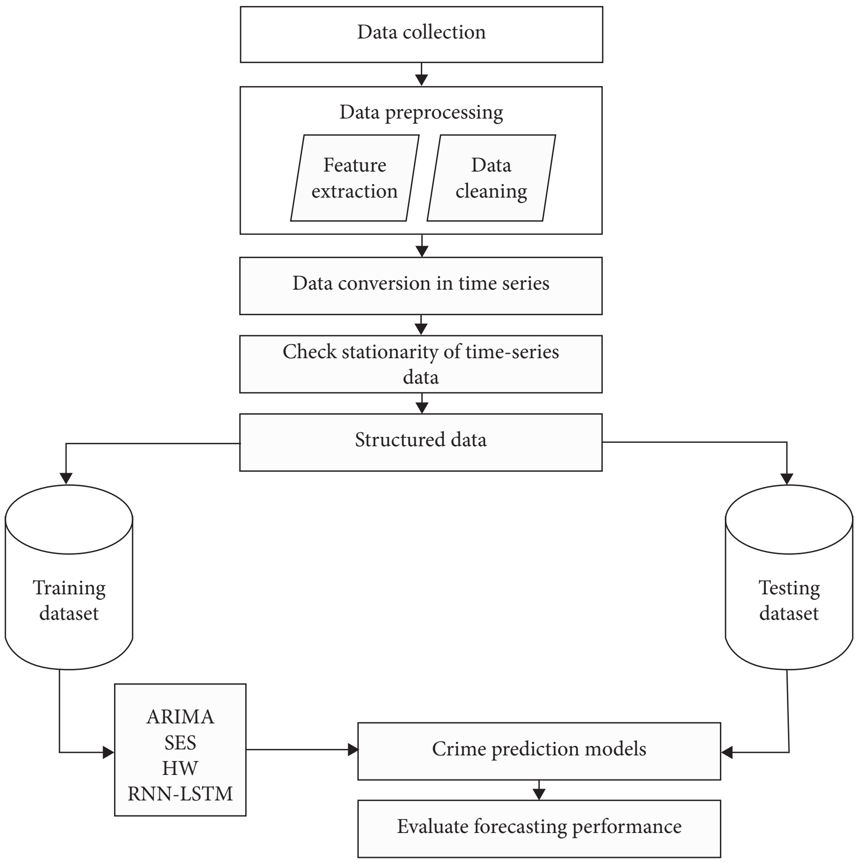


Figure 3: Proposed methodology.

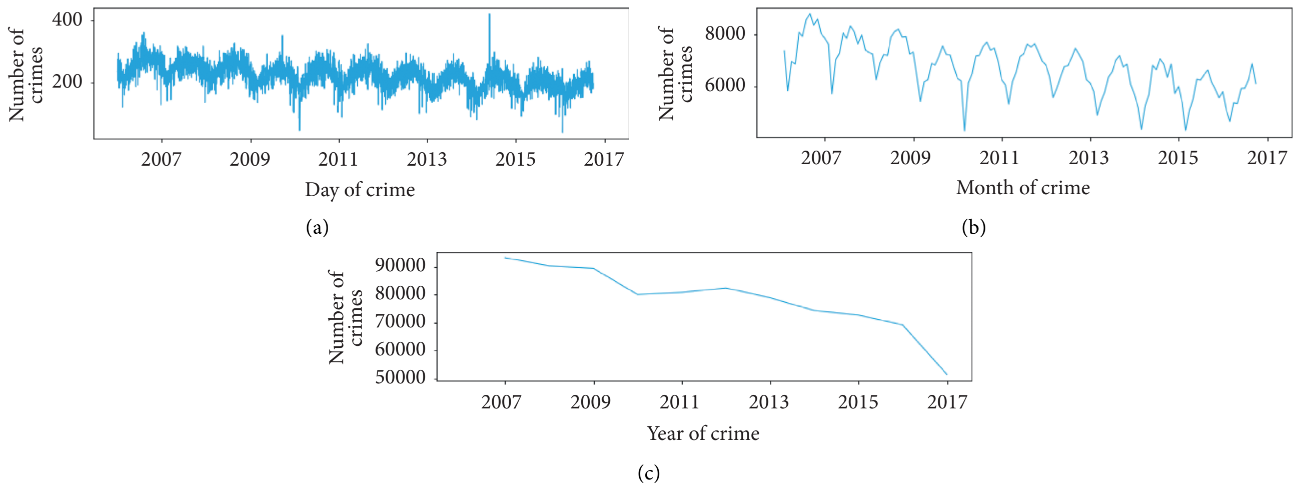


Figure 4: (a) Daily violent crimes, (b) monthly violent crimes, and (c) yearly violent crimes from 2006 to 2016.

crime occurrences on the Y-axis. The crime rate was gradually decreasing from 2007 to 2010. However, from 2011 to 2016, the violent crime was oscillating. Figure 4(c) shows the violent crime has a downward trend from the observed data by year (2006 to 2016). From this insight, it is hypothesized that the raw visualization of these data is distributed evenly over the days, but the trend is going down in monthly and yearly data. Crime occurrences by day, month, and year have a clear trend along with seasonal variations in a dataset. There are many variations present in the daily data; thus, crime data are resampled in monthly data to apply the time series algorithms.

For time series analysis, data must be in a stationary form which means the series should be without trend and with constant variance, mean, and covariance over time. If the data are in the nonstationary form, then they are unpredictable and cannot be forecasted. Stationary data should have a constant mean, variance, and covariance over time.

The data exploration shows that crime data are nonstationary. Therefore, the data need to be converted into a stationarity form to forecast the crimes.

Data were made stationary using rolling statistics mean and augmented Dickey–Fuller (ADF). Rolling statistics mean can be applied to moving mean or moving standard deviation at any instant time t. This was applied for the t-statistic, p value, lags, and the number of observations. The differencing technique is employed through first-order differencing ($d = 1$) on crime data to make data stationary on mean to remove the trend. The variance of the data should also be stationary to obtain reliable forecasts using different forecasting models. This test identifies whether the data consist of a unit root feature that has a severe impact on statistical inference. The unit root test determines the strength of a time series by trend. Many actual datasets are too complex to be captured by simple autoregressive models. Dickey–Fuller test is built on linear regression and is the easiest way to detect the unit root.

In this study, the ADF test is applied to the raw crime data. We applied the difference of lag 1 on the raw crime data where the series is not having a longer trend. Moreover, a difference of lag 12 on raw data is applied to see the trend by removing the seasonality. A double-differencing technique is used in which the series is differenced by lag 12 and then differenced by lag 1. This gives us a double-difference series where there is no trend and seasonality.

The data should also be stationary on the variance to obtain reliable forecasts using ARIMA, SES, and HW models. Therefore, the logarithm is taken to transform the data to make them stationary on variance and to evaluate the influence of seasonality. The resultant integrated part of ARIMA, SES, and HW is equal to one as the first difference makes the series stationary.

Autocorrelation function (ACF) describes the correlation between lagged values of any series at different times [37]. ACF depicts the relationship between the present and past values of the series. It considers time series components such as seasonality, trend, cyclic, and residual to find correlations.

Next, ACF is plotted on stationary crime data to identify the presence of AR and MA components in the residuals (Figure 5). ACF values are shown on a vertical axis which ranges from -1 to 1. The horizontal axis illustrates the size of the lag between the elements of the time series. Daily and monthly ACF sample patterns determine the summarized model processes. The lag refers to the correspondence order. In the daily ACF plot at lag 0, the correlation is 1. The reason is the data are correlated with themselves. At a lag of 1, the correlation is approximately -0.4. There are enough dotted horizontal lines present that conclude residuals are not random. There is a seasonal component present in the residuals at the lag of 12, and information available can be extracted by AR and MA models.

The prediction methods were applied on univariate data that require the least number of observations prior to starting the models such as ARIMA, SES, HW, and RNN-LSTM. For the parametrized ARIMA model, there are three distinct integers (p, d, q), where p is for the AR model, q is for the MA model, and d is for an integrated part. In the ARIMA, a model fits the importance of the parameters with a certain number of parameters and tests. This means whether the parameters are expressed in unit roots (null hypothesis) or not (alternative hypothesis). The standards such as t-statistics and P value are used to evaluate the importance of the parameters considered for the model [38]. We have used $(p = 1, q = 1, d = 0)$ to fit the ARIMA model based on ACF results. In SES, the values of the data series are analyzed without trends and seasonality. The stationary data have been used to apply on SES. On the contrary, HW data values are forecasted with trends as well as seasonality. There occur some significant jumps in a few successive time points. After applying SES and HW, the amplitude of fluctuations varies based on the nature of data [39].

Lastly, LSTM is employed along with the RNN as the building unit or extension of the RNN. LSTM can read, write, and delete information or can retain information in its memory. RNN is applied along with LSTM to avoid the exploding gradient and vanishing gradient problem. RNN uses short-term memory where LSTM is working like a gated cell in the form of sigmoid ranging from 0 to 1 which can help backpropagation and keep the gradient steep, so the training is short and accuracy is high. RNN is used to handle the sequence-dependent variables of daily and monthly violent crimes. The normalization technique is applied to the data to make them uniform. LSTM for regression with time steps is applied on the violent crime in which the previous time step is taken in the series as the input to forecast the output at the next time step. This process is applied by setting the columns to be time-step dimension and changing the values of dimension back to 1. In this method, mapping is applied by finding the end of the data pattern, checking the limits of sequence, and gathering input and output parts of the pattern. Then, reshaping is done by taking the current time $(x = t)$ which is going to predict the value at the next time in the sequence $(y = t + 1)$. The network is trained with 100 epochs, 1 batch size, and 2 verbose.

5. Results and Discussion

The time-series prediction techniques have been applied and compared to evaluate the effectiveness and efficiency. In order to perform regression tasks and their validation, the crime data are divided into training and testing data. This study is conducted by using a univariate data structure where UCR_General is the variable used against Dispach_Date_Time. UCR_General is the criminal code that is classified into violent crime and property crime.

There exist several ways to measure the accuracy of the forecasting method. For the regression problem, MAPE (equation (4)) and RMSE (equation (5)) are used as error metric measurements. Both MAPE and RMSE are used to evaluate modeling capabilities as well as predictive ability. Any forecast with the MAPE value ≤10% is observed as highly accurate. The value between 10 and 20% is considered good, while 21−50% is supposed to be reasonable. The value greater than 50% is considered inaccurate forecasting [40]. In this study, the MAPE value obtained is less than 10%. The

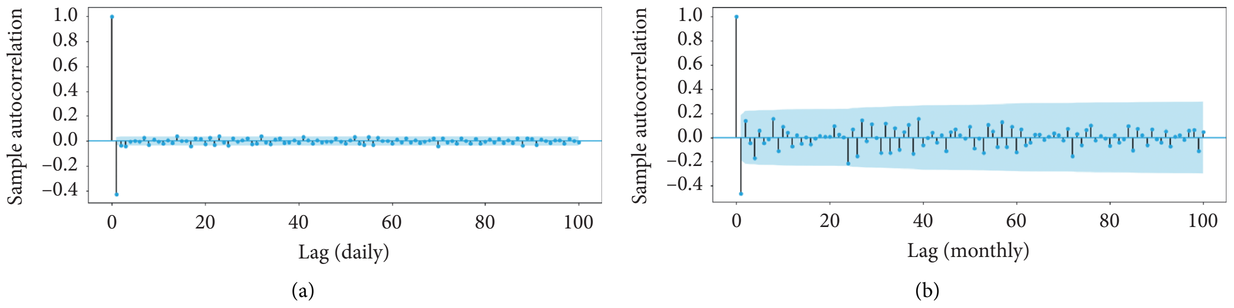


FIGURE 5: (a) Daily ACF and (b) monthly ACF in time series.

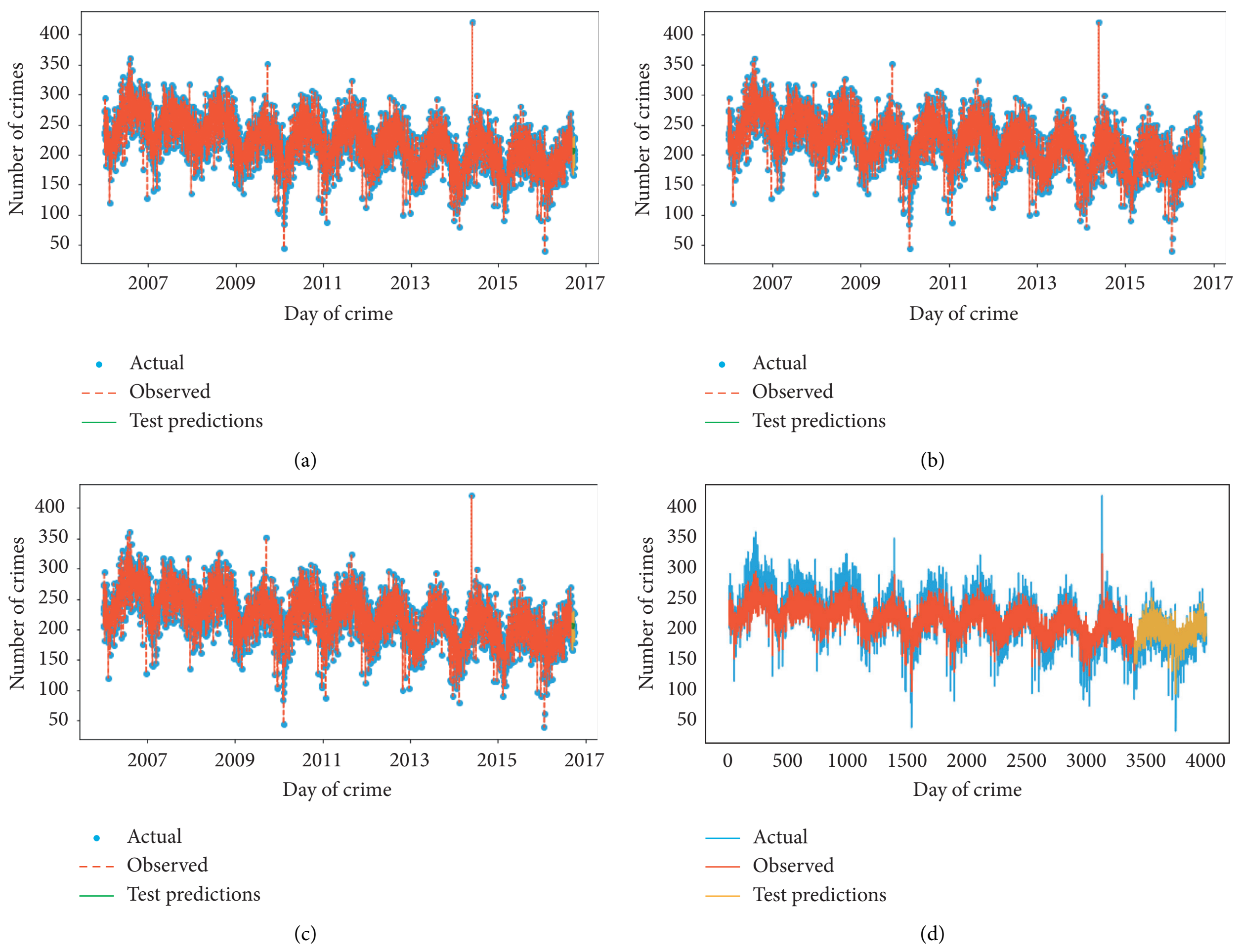


FIGURE 6: Actual and observed values for daily violent crimes by different models. (a) ARIMA model. (b) HW model. (c) SES model. (d) RNN model.

RMSE value can range from 0 to +∞, where 0 is the best value and indicates no difference between the values of the modeled and observed data.

$$\text{MAPE} = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{(a_t - \hat{a}_t)}{a_t}\right|, \tag{4}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(a_t - \hat{a}_t)^2}. \tag{5}$$

The violent crime data are analyzed in different ranges and periods for different models used in this study

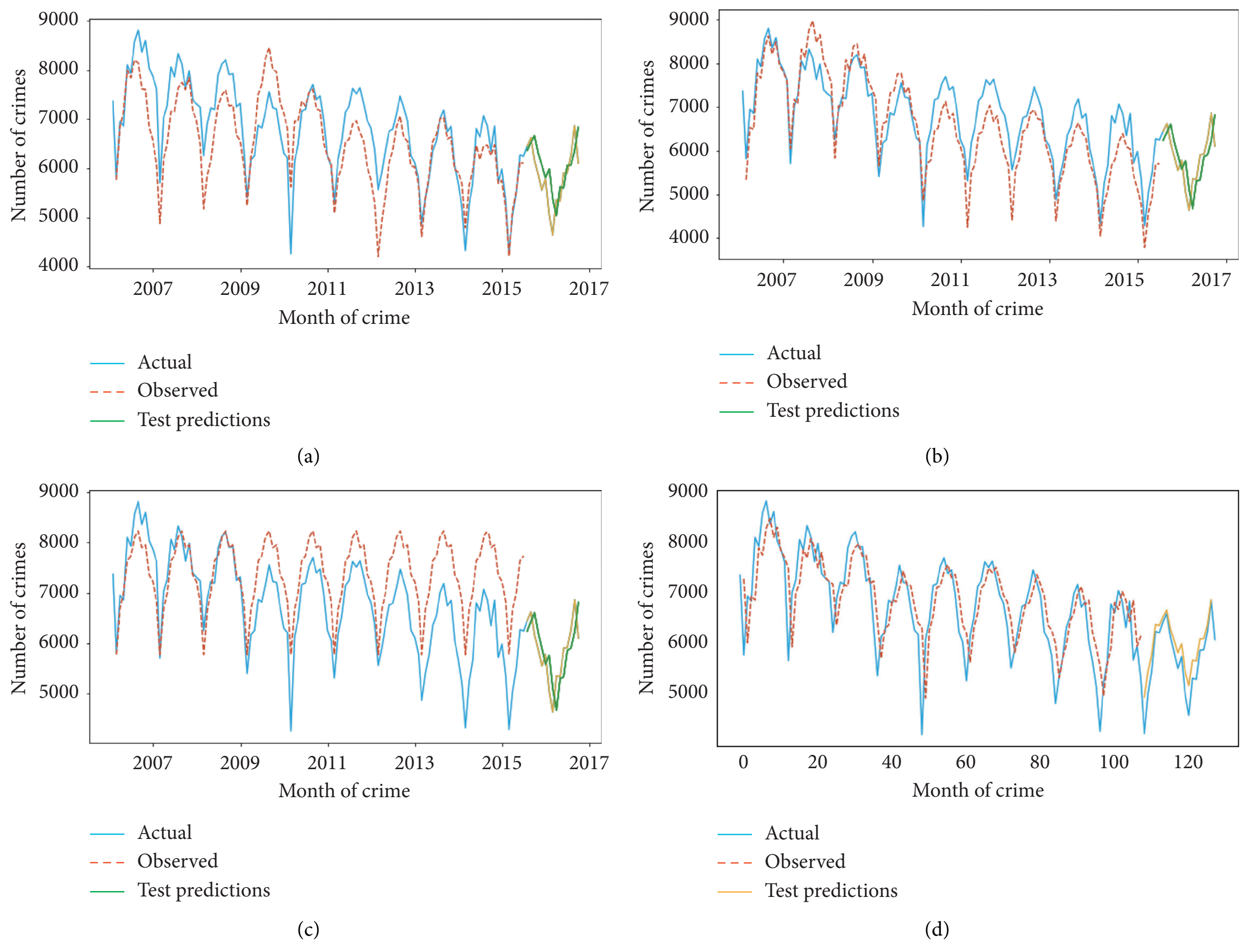


Figure 7: Actual and observed values for monthly violent crimes by different models. (a) ARIMA model. (b) HW model. (c) SES model. (d) RNN model.

Table 1: Error metric measurements: a performance comparison of daily and monthly violent crimes.

Model	Daily violent crime		Monthly violent crime	
	RMSE	MAPE	RMSE	MAPE
ARIMA model	201.96	81.84	583.09	6.90
HW model	40.60	17.39	536.51	7.20
SES model	39.60	16.37	973.01	13.28
RNN-LSTM model	13.42	4.75	18.42	6.03

(Figures 6 and 7). Each graph represents the number of crime events related to a particular aspect. The trends depict the actual and expected values for daily and monthly crimes by using different time series models. Figure 6 shows the fluctuated series obtained for crimes through a different model. This figure demonstrates the original and predicted values for daily violent crimes by using different models. There are a number of offenses in differing amounts and intervals of time. The violent crime increases in the middle of the day and descends in the evening of the day. There is a downward trend component in daily crimes from 2013 to 2016 period.

Figure 7 depicts the result of the monthly violent crime using different models. There is a series of offenses to different extents and time frames. Violent incident headlines are made on a regular basis in Philadelphia, and the violent crime spikes in summer [28]. The crimes ascend in months of summer (June, July, and August) and descend in months of winter. There is a downward trend component in monthly crimes that have come down around 2016.

Table 1 provides more details about error metrics for daily and monthly crimes. RNN-LSTM has much better performance than the ARIMA, SES, and HW for daily and monthly crime forecast. LSTM using RNN has a higher forecasting accuracy and the lower gap from other models between training and testing errors. The proposed method is useful and can be easily applied to the time-series regression problems.

6. Conclusion

The purpose of this study was to develop a time series model using statistical model experimentation and predict the daily and monthly violent crime in Philadelphia. This study performs the comparative analysis of predictive models based on RMSE and MAPE values. RNN-LSTM has achieved better results than the other models with the values of RMSE 4.75 and MAPE 13.42. The RNN-LSTM model has a higher forecasting accuracy and a lower gap from other models between training and testing errors. These results can help law enforcement agencies in decision-making. In the future, we are interested to develop specific recommendations or targeted crime prevention strategies for different crime prevention models. Moreover, we will perform the scalability analysis and implement the proposed method for different datasets.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Q. Wang, G. Jin, X. Zhao, Y. Feng, and J. Huang, "CSAN: a neural network benchmark model for crime forecasting in spatio-temporal scale," *Knowledge-Based Systems*, vol. 189, Article ID 105120, 2020.

[2] H. Borrion, P. Ekblom, D. Alrajeh et al., "The problem with crime problem-solving: towards a second generation pop?" *The British Journal of Criminology*, vol. 60, no. 1, pp. 219–240, 2020.

[3] "The texas crime report for 2017," 2017, https://www.dps.texas.gov/crimereports/17/citCh2.pdf.

[4] "Violent crime - wikipedia," 2016, https://en.wikipedia.org/wiki/Violentcrime.

[5] "Fbi–violent crime," 2011, https://ucr.fbi.gov/crime-in-the-u.s/2011/crime-in-the-u.s.-2011/violent-crime/violent-crime.

[6] E. Cesario, C. Catlett, and D. Talia, "Forecasting crimes using autoregressive models," in *Proceedings of the IEEE 14th International Conference on Dependable, Autonomic and Secure Computing*, pp. 795–802, IEEE, Sydney, Australia, 2016.

[7] C. L. J. Douglas, C. Montgomery, and M. Kulahci, *Introduction to Time Series Analysis and Forecasting*, Wiley Online Library, Hoboken. NJ, USA, 2015.

[8] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the arima model," in *Proceedings of the 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pp. 106–112, IEEE, 2014.

[9] H. Yu, L. J. Ming, R. Sumei, and Z. Shuping, "A hybrid model for financial time series forecasting-integration of ewt, arima with the improved abc optimized elm," *IEEE Access*, vol. 8, pp. 84 501–584 518, 2020.

[10] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[11] C. Deb, F. Zhang, J. Yang, S. E. Lee, and K. W. Shah, "A review on time series forecasting techniques for building energy consumption," *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 902–924, 2017.

[12] Ď. Peter and P. Silvia, "Arima vs. arimax–which approach is better to analyze and forecast macroeconomic time series," in *Proceedings of the 30th International Conference Mathematical Methods in Economics*, pp. 136–140, Karviná, Czech Republic, 2012.

[13] M. Ahmad, A. Khan, M. Mazzara, and S. Distefano, "Seeking optimum system settings for physical activity recognition on smartwatches," in *In Advances in Computer Vision*, K. Arai and S. Kapoor, Eds., pp. 220–233, Springer International Publishing, Cham, Switzerland, 2020.

[14] M. Ahmad, M. Alqarni, A. Khan et al., "Smartwatch-based legitimate user identification for cloud-based secure services," *Mobile Information Systems*, vol. 2018, Article ID 5107024, 14 pages, 2018.

[15] M. Ahmad, A. M. Khan, M. Mazzara, S. Distefano, A. Ali, and A. Tufail, "Extended sammon projection and wavelet kernel extreme learning machine for gait-based legitimate user identification," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pp. 1216–1219, ACM, New York, NY, USA, 2019.

[16] C. D. R. Rodríguez, D. M. Gomez, and M. A. M. Rey, "Forecasting time series from clustering by a memetic differential fuzzy approach: an application to crime prediction," in *Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, IEEE, Honolulu, HI, USA, 2017.

[17] P. Chen, H. Yuan, and X. Shu, "Forecasting crime using the arima model," in *Proceedings of the 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 5, pp. 627–630, IEEE, Jinan, China, 2008.

[18] H. B. Hwarng and H. Ang, "A simple neural network for arma time series," *Omega*, vol. 29, no. 4, pp. 319–333, 2001.

[19] P. S. Kalekar, "Time series forecasting using holt-winters exponential smoothing," *Kanwal Rekhi School of Information Technology*, vol. 4329008, no. 13, 2004.

[20] E. Ostertagova and O. Ostertag, "Forecasting using simple exponential smoothing method," *Acta Electrotechnica et Informatica*, vol. 12, no. 3, p. 62, 2012.

[21] E. Ostertagová and O. Ostertag, "The simple exponential smoothing model," *Proceedings of Conference*, Technical University of Košice, in *Proceedings of the 4th International Conference on Modelling of Mechanical and Mechatronic Systems*, pp. 380–384, Technical University of Košice, Košice, Slovak Republic, 2011.

[22] G. Tirkes, C. Guray, and N. Celebi, "Demand forecasting: a comparison between the holt-winters, trend analysis and decomposition models/predvidanje potraznje: usporedba izmedu holt-winters modela, analize trenda i modela dekompozicije," *Tehnicki Vjesnik-Technical Gazette*, vol. 24, no. S2, pp. 503–510, 2017.

[23] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[24] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[25] A. Stec and D. Klabjan, "Forecasting crime with deep learning," 2018, http://arxiv.org/abs/1806.01486.

[26] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep lstm-rnn," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2727–2740, 2019.

[27] T. Almanie, R. Mirza, and E. Lor, "Crime prediction based on crime types and using spatial and temporal criminal hotspots," 2015, http://arxiv.org/abs/1508.02050.

[28] J. Ratcliffe, R. Taylor, A. Askey, J. Grasso, and R. Fisher, *The Philadelphia Predictive Policing Experiment: Impacts of Police*

Cars Assigned to High Crime Grids, Center for Crime Science, Temple University, Philadelphia, PA, USA, 2017.

[29] C. S. Marzan, M. J. C. Baculo, R. de Dios Bulos, and C. Ruiz, “Time series analysis and crime pattern forecasting of city crime data,” in *Proceedings of the International Conference on Algorithms, Computing and Systems*, pp. 113–118, ACM, Beijing, China, 2017.

[30] G. Borowik, Z. M. Wawrzyniak, and P. Cichosz, “Time series analysis for crime forecasting,” in *Proceedings of the 2018 26th International Conference on Systems Engineering (ICSEng)*, pp. 1–10, IEEE, Sydney, Australia, 2018.

[31] M. Feng, J. Zheng, Y. Han, J. Ren, and Q. Liu, “Big data analytics and mining for crime data analysis, visualization and prediction,” in *Proceedings of the International Conference on Brain Inspired Cognitive Systems*, pp. 605–614, Springer, Guangzhou, China, 2018.

[32] P. Singh, *Time series forecasting on crime data in amsterdam for a software company*, Ph.D. dissertation, NOVA Information Management School, Instituto Superior de Estatística e Gestão de Informação, Universidade Nova de Lisboa, Lisbon, Portugal, 2018.

[33] C. Catlett, E. Cesario, D. Talia, and A. Vinci, “Spatio-temporal crime predictions in smart cities: a data-driven approach and experiments,” *Pervasive and Mobile Computing*, vol. 53, pp. 62–74, 2019.

[34] Metadata catalog – phila.Gov,” https://metadata.phila.gov/home/datasetdetails/5543868920583086178c4f8e/representationdetails/570e7621c03327dc14f4b68d/.

[35] S. Deegalla, H. Boström, and K. Walgama, “Choice of dimensionality reduction methods for feature and classifier fusion with nearest neighbor classifiers,” in *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pp. 875–881, IEEE, Chicago, IL, USA, 2012.

[36] S. Ahmadkhani and P. Adibi, “Face recognition using supervised probabilistic principal component analysis mixture model in dimensionality reduction without loss framework,” *IET Computer Vision*, vol. 10, no. 3, pp. 193–201, 2016.

[37] J. D. Cryer, *Time Series Analysis*, vol. 286, Duxbury Press, Boston, MA, USA, 1986.

[38] Y. S. Triana and A. Retnowardhani, “Enhance interval width of crime forecasting with arima model-fuzzy alpha cut,” *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 3, pp. 1193–1201, 2019.

[39] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, OTexts, Melbourne, Australia, 2018.

[40] J. J. M. Moreno, A. P. Pol, A. S. Abad, and B. C. Blasco, “Using the r-mape index as a resistant measure of forecast accuracy,” *Psicothema*, vol. 25, no. 4, pp. 500–506, 2013.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6654926, 7 pages
https://doi.org/10.1155/2021/6654926

Research Article

Cuckoo Search-based SVM (CS-SVM) Model for Real-Time Indoor Position Estimation in IoT Networks

Amjad Khan,[1] Asfandyar Khan,[1] Javed Iqbal Bangash,[1] Fazli Subhan,[2] Abdullah Khan,[1] Atif Khan,[3] M. Irfan Uddin,[4] and Marwan Mahmoud[5]

[1]*Institute of Computer Sciences and Information Technology (ICS/IT), The University of Agriculture, Peshawar 25000, Pakistan*
[2]*Department of Computer Science, National University of Modern Languages (NUML), Islamabad 44000, Pakistan*
[3]*Department of Computer Science, Islamia College Peshawar, Peshawar 25000, Pakistan*
[4]*Institute of Computing, Kohat University of Science and Technology, Kohat, Pakistan*
[5]*Faculty of Applied Studies, King Abdulaziz University, Jeddah, Saudi Arabia*

Correspondence should be addressed to Atif Khan; atifkhan@icp.edu.pk

Received 13 December 2020; Revised 13 January 2021; Accepted 8 March 2021; Published 19 March 2021

Academic Editor: Shah Nazir

Internet of Things (IoT), an emerging technology, is becoming an essential part of today's world. Machine learning (ML) algorithms play an important role in various applications of IoT. For decades, the location information has been extremely useful for humans to navigate both in outdoor and indoor environments. Wi-Fi access point-based indoor positioning systems get more popularity, as it avoids extra calibration expenses. The fingerprinting technique is preferred in an indoor environment as it does not require a signal's Line of Sight (LoS). It consists of two phases: offline and online phase. In the offline phase, the Wi-Fi RSSI radio map of the site is stored in a database, and in the online phase, the object is localized using the offline database. To avoid the radio map construction which is expensive in terms of labor, time, and cost, machine learning techniques may be used. In this research work, we proposed a hybrid technique using Cuckoo Search-based Support Vector Machine (CS-SVM) for real-time position estimation. Cuckoo search is a nature-inspired optimization algorithm, which solves the problem of slow convergence rate and local minima of other similar algorithms. Wi-Fi RSSI fingerprint dataset of UCI repository having seven classes is used for simulation purposes. The dataset is preprocessed by min-max normalization to increase accuracy and reduce computational speed. The proposed model is simulated using MATLAB and evaluated in terms of accuracy, precision, and recall with K-nearest neighbor (KNN) and support vector machine (SVM). Moreover, the simulation results show that the proposed model achieves high accuracy of 99.87%.

1. Introduction

Internet of Things (IoT) is an emerging technology that provides different devices to interconnect and communicate with each other. IoT is becoming an important part of today's world due to its rapid growth. Moreover, the use of machine learning (ML) algorithms in various applications of IoT has attracted researchers from all over the world. For a very long time, location has been extremely useful for humans to navigate outdoor over the sea, air, and land using astrolabe, sextant, and octant to determine their location with respect to various celestial bodies [1]. In the 20th century, with the advancement in electronics and communication, new technologies are adapted such as Radio Detection and Ranging (RADAR), Long Range Navigation (LORAN), and Global Positioning System (GPS) for localization [1]. GPS remains one of the most dominant technologies among the available technologies to localize an object. It only shows better performance to localize object outdoor and fails to estimate the position of object indoor with acceptable accuracy. Now, people are spending most of their time in an indoor environment, thus needing the positioning system to trace people and objects in the indoor complex environment. Therefore, many applications have been arised, which need location information such as location detection of products in a warehouse, location detection of personal in hospitals, and localizing fireman in a building.

To estimate the position of an object either outdoor or indoor, the most usable and powerful technique used was the global positioning system (GPS). GPS estimates the location by measuring the distance between a GPS satellite and a base station using LoS. The GPS-based location estimation techniques fail to achieve high accuracy due to high signal loss inside a complex indoor environment as GPS signals cannot penetrate the walls of buildings and other obstacles [2]. Due to the technological advancement, many other signal-based possibilities have been raised such as camera, sound, infrared, Radio Frequency Identification (RFID) and Bluetooth Tags, and Wi-Fi [2]. Among all these Wi-Fi received more attention from the research community because, in most cases, the site is already calibrated with Wi-Fi routers, which obsolete extra calibration charges and time [3]. Different techniques are developed, such as Triangulation, Trilateration, Proximity, and Fingerprinting, using Angle of Arrival (AoA), Time difference of Arrival (TDoA), Time of Arrival (ToA), and Receive Signal Strength Identification (RSSI) [1]. All these techniques except fingerprinting require LoS, which is not possible in an indoor environment which makes fingerprinting the most reasonable technique for indoor localization [4]. On the contrary, fingerprinting is laborious and time-consuming and the radio map is venerable to environmental changes, leading to high position estimation error. Machine learning-based models are introduced to automate, generalize, and reduce estimation error [5].

Many machine learning algorithms such as support vector machine (SVM), K-nearest neighbor (KNN), extreme learning model (ELM), decision tree (DT), Naive Bayes (NB), and Bayesian Network (BN) were used for location estimation in an indoor environment. The results show that KNN and SVM are outperformers [6, 7] as compared to others. Moreover, SVM is based on the structural risk minimization principle with good generalization ability and can better solve problems with few samples, nonlinear data, avoid local minima, and so on [2]. For high classification accuracy or position estimation machine learning models, SVM depends on their parameter optimization. Therefore, nature-inspired optimization algorithms such as particle swarm, bee, bad reference distribution and cuckoo search can be used [8].

Cuckoo is one of the most recent algorithms inspired by breeding phenomena of the cuckoo bird, which are used to solve the nonlinear optimization problem. Other optimization algorithms have limitations in terms of convergence to the current or local best solution. They may fail to solve the nonlinear optimization or multidimensional optimization problem. In the case of cuckoo search, combining local and search capability increases the probability of global optimal solution using Levy's flight process [9].

In this study, we propose a cuckoo search-based support vector machine (CS-SVM) model for position estimation in an indoor complex environment. Inspired by many state-of-the-art optimization-based machine learning models, we used a state-of-the-art dataset of the well-known UCI repository, which is the same as in [10], to evaluate its performance. The proposed model is evaluated in terms of accuracy, precision, and recall with KNN and SVM using MATLAB. The KNN and SVM stay good performers achieving room level accuracy up to 98.7% and 98.3%, respectively, while the proposed model achieves high accuracy up to 99.7%.

In Section 1, we elaborated the literature study, and then, in Section 2, the ingredients of the proposed model are discussed along with the proposed model, and in Section 3, results of the proposed model are justified with benchmark results. Section 4 concludes the research article.

2. Related Work

In [11], the authors conducted a survey regarding localization techniques, mentioned that GPS and cellular networks are outdoor localization sources, and they failed to localize anything indoor because of the deep shadowing effect. In [12], Subhan et al. proposed an extended gradient predictor and filter to reduce variation in RSSI values. The RSSI values get variation due to various factors such as walls, obstacles, human crowd, and temperature. The results show better performance than the KALMAN filter. On the contrary, Suining and Chan [3] proposed a fingerprinting technique and used Wi-Fi RSSI values to reduce the extra calibration expenses.

Recently, Wi-Fi RSSI is used to estimate the position of an object in an indoor environment. In [4], Wi-Fi-based approach is proposed using two architectures: client server and standalone. It uses the existing infrastructure of an indoor environment and compares offline fingerprint RSSI measurement with an online RSSI fingerprint to estimate the location of the user. A combination of Wi-Fi and Bluetooth radio technology-based approach is proposed in [13]. It uses KNN with particle filter and shows that indoor estimation error changes by changing the target area. Hossain and Soh [5] highlights that Wi-Fi fingerprinting is laborious and time-consuming, and radio maps are vulnerable to environmental changes. In [14], the authors proposed a multidimensional particle filter (MPF) algorithm to estimate the direction of an indoor object. The scheme in [15] is based on Bluetooth technology and uses a machine learning approach to automate the fingerprinting technique. The RSSI variations are smoothened using the filtering algorithm to achieve high accuracy. In [16], the authors presented a learning regression-based filter tracking system using RSSI matrices. It concludes that the particle filter is efficient for the loud and complex indoor environment but expensive than the KALMAN filter.

In [6], the authors compare various machine learning algorithms such as KNN, SVM, NB, BN, DT, and SMO and ensemble algorithms such as Bagging and AdaBoost using fingerprinting technique. The simulation results show that KNN is the best of all. Similarly, Sabanci et al. [7] also compare different machine learning algorithms such as ANN, KNN, ELM, SVM, NB, and DT based on Wi-Fi fingerprinting. According to the simulation results, the KNN shows the best performance. In [17], the KALMAN filter is used to smooth RSSI values coming from Bluetooth beacon and compare KNN, SVM, and random forest. In [18], it is stated that fingerprinting map changes with change in the environment which leads to high positioning estimation error. For this purpose, KNN, SVM, and DT techniques are used.

Other researchers have also used machine learning algorithms such as KNN and WNN [19], ELM [20], SVM [7], and SVM and DT [21] for position estimation in an indoor environment. According to the literature two machine learning algorithms, both K-NN and SVM show better performance against the other learning model. Compared to SVM, KNN slightly shows good accuracy results in the literature cited.

2.1. Proposed Methodology. In the following sections, the components of the proposed model (CS-SVM), i.e., support vector machine (SVM) and cuckoo Search along with the proposed CS-based SVM, are discussed.

2.2. Cuckoo Search Algorithm. The cuckoo search algorithm [9] was developed inspired by the breeding process of cuckoo birds. Recently, gaining more attention and becoming very popular over other optimization algorithms such as particle swarm, bat, and hill climbing, these algorithms are also nature-inspired, but they have the limitation of converging to the local or current best solution. So, they are lack in their performance for a nonlinear and multidimensional optimization problem. On the contrary, cuckoo search adopts a different strategy to best fit for the multidimensional and nonlinear problem. It uses Levy's flight process, where the selection of local best through searching capability gives the high chance of global optimal solution. The cuckoo search working process is discussed in the following steps.

At a time, a cuckoo lays only one egg, and the eggs are placed in a nest, which is selected randomly. In a nest, the different eggs represent different solutions, while the new solution is represented by the cuckoo egg.

The nests having high-quality eggs are the best nests, which will be passed to the next generation. The best solutions are represented by these best nests.

There is a fixed number of host nests that are available. $P \epsilon (0,1)$ is the probability which represents that the egg laid by a cuckoo is revealed by the host bird. Accordingly, Levy flight mechanism equation (1) is used to estimate the updated nest position of the cuckoo:

$$X_i^{t+1} = X_i^{(t)} + \partial \oplus L(\lambda), \quad i = 1, 2, \ldots, n. \tag{1}$$

In equation (1), $X_i^{(t)}$ represents the position of the nest, X_i^{t+1} is the new nest position, ∂ is the control value, and $\oplus$ represents point to point multiplication of the $L(\lambda)$ Levy flights' process. After updating, the position random value is generated, where $r\,\varepsilon\,(0,1)$.

If $r > P$, new position X_i^{t+1} changes randomly or it remains in the same position, and the better nest with the new position X_i^{t+1} is kept for the next generation.

2.3. Support Vector Machine. Support vector machine (SVM) can be used for classification as well as a regression problem. It works on the principle of structural risk minimization (SRM). It balances the linear separable space data into nonlinear separable feature space. Equation (2) gives the linearly separable sample set for the binary classification problem:

$$(x_i, y_i), \; i = 1, 2, \ldots, n, \; x \in R^d. \tag{2}$$

In case x_i belongs to the first class, then it is denoted by $y_i = 1$, while in case x_i is belongs to the second class, then it is denoted by $y_i = -1$.

Here, the division line called hyperplane classifies two classes without error, a margin line which specifies class boundary and the distance between the margin lines of two opposite classes called class interval or marginal distance and the data point of either class. The line which is nearest to the hyperplane is known as the support vector. In the high-dimensional space, the marginal distance makes the hyperplane more optimal which results in the optimal division line into the optimal division plane. Radial basis function (RBF), sigmoid kernel, polynomial kernel, and linear kernel are the commonly used kernels. Considering practical applications, the classification problems belong to multiclass category problems. The indoor positioning problem with multiple class dataset belongs to multiclass category problems. Therefore, the establishment of an SVM multiclassifier is required. Directed acyclic graph, one-versus-all, and one-versus-one are multiclassification methods.

2.4. Cuckoo Search-Based SVM (CS-SVM). Appropriate parameter selection is very important as both the generalization and learning performance ability of SVM depend on it. Moreover, the perdition ability of the model and its precision has a direct relation with the appropriate parameter selection. Therefore, the SVM parameters can be optimized using different methods such as grid search method, genetic algorithm, and particle swarm optimization algorithm. Both the genetic and particle swarm optimization algorithms face the problem of local extremes. On the contrary, the grid search method is time-consuming as over the hyperparameter space an exhaustive search is required. Recently, the cuckoo search (CS) algorithm is proposed which is a metaheuristic algorithm. It has a strong ability to global search, requires fewer parameters, and has a good search path. To solve those problems, having multiobjective is a powerful tool. The flow of the proposed is shown in Figure 1.

The performance of the SVM classifier is dependent mainly on the kernel parameter σ and the penalty factor C. The following steps are used to optimize the SVM parameters, which are also shown in Figure 2:

(1) Training dataset is selected to train the SVM.

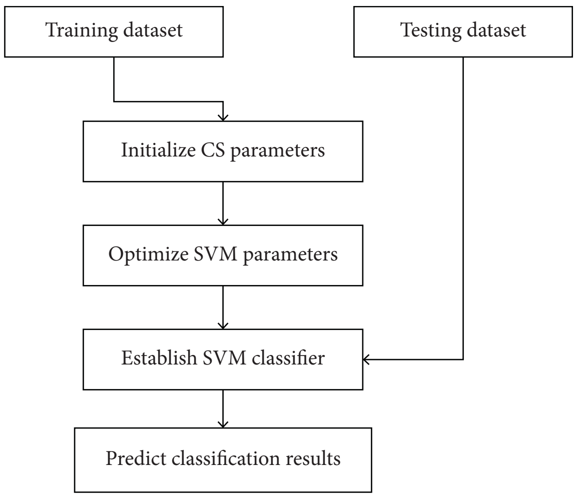


Figure 1: Architecture of the proposed model.

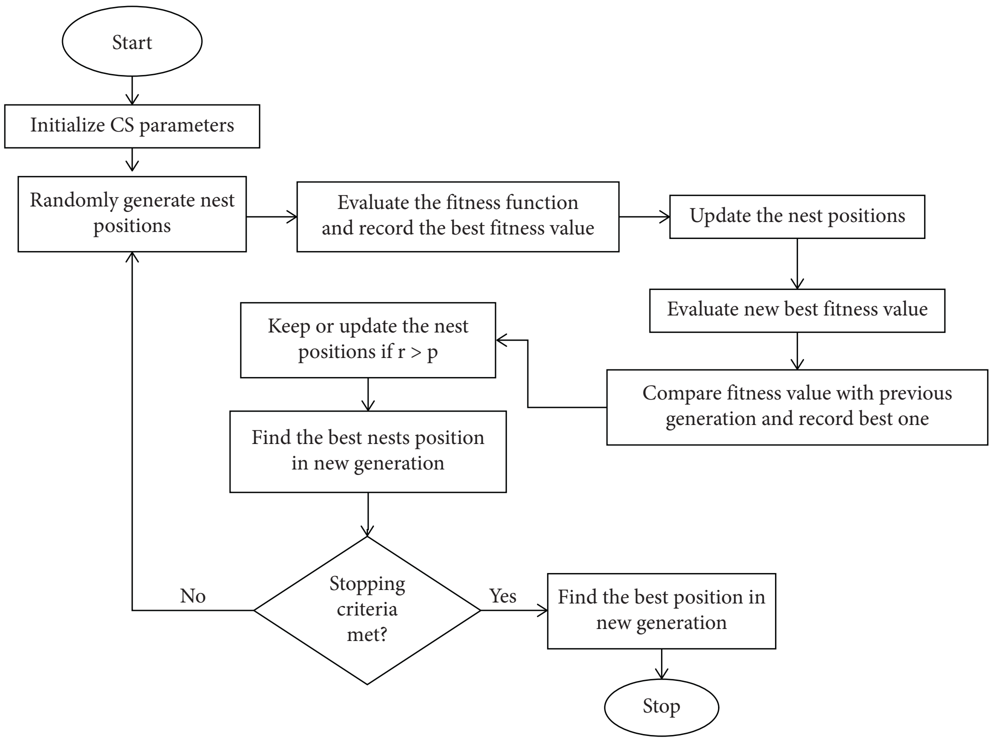

Start
Initialize CS parameters
Randomly generate nest positions
Evaluate the fitness function and record the best fitness value
Update the nest positions
Evaluate new best fitness value
Keep or update the nest positions if r > p
Compare fitness value with previous generation and record best one
Find the best nests position in new generation
No
Stopping criteria met?
Yes
Find the best position in new generation
Stop

Figure 2: Flow diagram of the cuckoo search-based SVM (CS-SVM) model.

(2) The CS parameters such as probability “P,” no. of nests “n,” the number of iteration, and SVM parameter ranges are initialized.

(3) Using solution space the initial population of ‘n’ host nests is generated randomly using

$$P_{t-1} = \left[x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}\right]^T. \quad (3)$$

After that, the eggs are placed there and the group of parameters (C, σ) represents the nest position.

(4) The qualities of the group of parameters (C, σ) which represents the nest positions (fitness functions) are evaluated to determine the current best nest $x_b^{(0)}$ (fitness value) and carry over it to the next generation.

(5) Equation (1) is used to update the positions of all other nests, and the qualities of the nest positions are evaluated belonging to the new group.

(6) The nest positions of this new group are compared with the last group using

$$p_{t-1} = \left[x_1^{(t-1)}, x_2^{(t-1)}, \ldots, x_n^{(t-1)}\right]^T. \quad (4)$$

Once the comparison is done, the group having the worse nest positions are replaced with the group having the better nest positions to get a group of better nest positions (fitness value) using

$$K_t = \left[x_1^{(t-1)}, x_2^{(t-1)}, \ldots, x_n^{(t-1)}\right]^T. \quad (5)$$

(7) In case r (random number) is greater than ″P, ″ keep the nest positions having low probability in k_t using equation (5), update the nest positions having a high probability, and evaluate the qualities of the nest positions belonging to the new group. The nest positions belonging to this group are compared with those in k_t. Once the comparison is done, the group having the worse nest positions is replaced with the group having the better nest positions to get a group of better nest positions (fitness value) using

$$p_t = \left[x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}\right]^T. \quad (6)$$

(8) Determine the best nest position (fitness value) $x_b^{(t)}$ in p_t using equation (6).

(9) Check whether the number of iterations has reached the threshold level of a number of iterations or the level of a certain precision has been achieved. In case none of the aforementioned conditions is true, go back to step (4) and continue. In case any one of the aforementioned conditions is true, stop searching, and $x_b^{(t)}$ is the best nest position.

(10) SVM parameters (C, σ) correspond to the best nest position $x_b^{(t)}$.

3. Results and Discussion

Inspired by the many state-of-the-art optimization-based machine learning models, we used a state-of-the-art dataset of the well-known UCI repository which is the same as in [10]. The dataset is preprocessed by min-max normalization to increase accuracy and reduce computational speed. It was divided into 70% training and 30% testing. The proposed model is simulated using MATLAB R 2018 b on Window 8 OS with 4 GB RAM.

Different training and testing experiments were performed on three models, i.e., support vector machine (SVM), K-nearest neighbor (KNN), and cuckoo search-based support vector machine (CS-SVM). These models were evaluated in terms of precision, recall, and sccuracy.

In the classification process with KNN, high accuracy values are achieved by optimizing the parameters. In machine learning (ML), the parameters that need to set the algorithm to start are known as hyperparameters. k and distance type to be calculated are hyperparameters in the case of KNN. As a result of the hHyperparameter optimization, the best k value is calculated as 1. The distance type that gives the best correctness is determined as Euclidean. Before training the KNN classifier, we divide the dataset into 70% training and 30% testing using holdout cross-validation. Test the trained KNN model over 600 observations, which are 30% of all total observations. Testing observation is distributed among the four classes. Each class is having 150 observations. The class-wise and average output prediction of the model in terms of precision, recall, and accuracy is shown in Table 1. The average precision, recall, and accuracy values are slightly different which are 0.987, 0.98675, and 0.98675, respectively. This result of the model showed slightly better performance than SVM.

Using an SVM classifier, first of all, the entire multiclass problem is converted into the binary class problems. The binary class problems are solved with binary classifiers, and the solution can be merged to get the solution of the multiclass problem. One-versus-one (OVO) method of SVM is used in such cases. In the OVO method, all possible combinations of the multiple class problems are divided into binary class problems. After that, the classifier is trained for each binary class problem. Then, the outputs of these binary class classifiers are merged to estimate the output multiple class problems. The SVM classification using the OVO method results in the error matrix. Likely, the SVM model tested was over 600 observations, which are 30% of all total observations. Testing observation is distributed among the four classes. Each class is having 150 observations. The class-wise and average output prediction of the model in terms of precision, recall, and accuracy is shown in Table 2. The average precision, recall, and accuracy values are slightly different from each other which are 0.98375, 0.98325, and 0.983, respectively. This result of the model is slightly behind the results of KNN.

From Tables 1 and 2, it is clear that SVM is slightly behind in their results against K-NN in this research experiment. Therefore, the CS-SVM trained model is tested over 600 samples of the dataset which is the same as the simple SVM and KNN. In this testing process, cuckoo search optimizes the parameter of SVM over the 6th iteration; the last iteration results are the final results of the CS-SVM model which are better than those of simple KNN and SVM, and now, SVM takes over the KNN results. The intended precision, recall, and accuracy results of the final iteration are 0.9900, 0.9980, and 0.9967, respectively, as shown in Table 3.

The performance of the proposed CS-SVM model in terms of precision, recall, and accuracy as compared to KNN and SVM is given in Table 4 and Figure 3. From Table 4 and Figure 3, it is clear that the proposed model CS-SVM surpasses the benchmark models.

According to the literature study and our implementation result, KNN gives a slightly better result than

Table 1: Precision, recall, and accuracy for KNN.

Classes	Precision	Recall	Accuracy
One	0.98	1.0	1.0
Two	1.0	0.967	0.967
Three	0.968	0.993	0.993
Four	1.0	0.987	0.987
Average	0.987	0.98675	0.98675

Table 2: Precision, recall, and accuracy for SVM.

Classes	Precision	Recall	Accuracy
One	1.0	0.993	0.993
Two	0.986	0.96	0.96
Three	0.949	0.987	0.986
Four	1.0	0.993	0.993
Average	0.98375	0.98325	0.983

Table 3: Precision, recall, and accuracy for CS-SVM.

Iteration	Precision	Recall	Accuracy
1	0.9950	0.9990	0.9983
2	0.9900	0.9980	0.9967
3	0.9950	0.9990	0.9983
4	0.9900	0.9980	0.9967
5	0.9900	0.9980	0.9967
6	0.9900	0.9980	0.9967

Table 4: Comparison of precision, recall, and accuracy of KNN, SVM, and CS-SVM.

Model	Precision	Recall	Accuracy
KNN	0.98375	0.98325	0.983
SVM	0.98375	0.98325	0.983
CS-SVM	0.9900	0.9980	0.9967

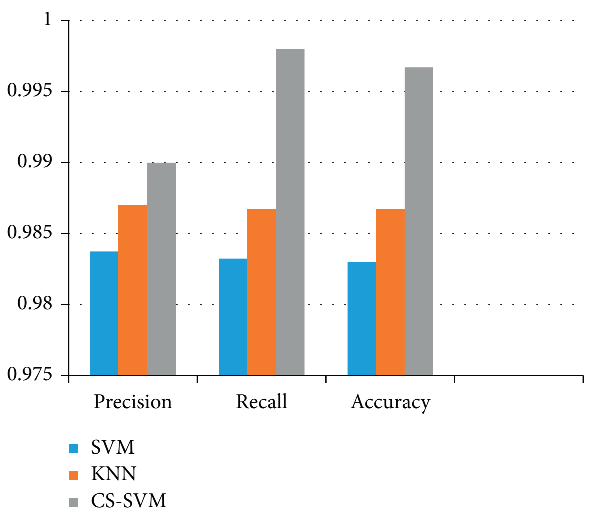


Figure 3: Precision, recall, and accuracy comparison of CS-SVM with SVM and KNN.

SVM. Now, by optimizing SVM through a nature-inspired evolutionary cuckoo search algorithm, the SVM improves results over KNN.

4. Conclusions

In this research work, we propose the cuckoo search-based support vector machine (CS-SVM) model for position estimation in an indoor complex environment. SVM is based on the structure risk minimization principle with good generalization ability and can better solve problems with few samples, nonlinear data, avoid local minima, and so on. Cuckoo is one of the most recent algorithms inspired by breeding phenomena of the cuckoo bird, which are used to solve the nonlinear optimization problem. Other optimization algorithms have limitations in terms of convergence to the current or local best solution. A state-of-the-art dataset of the well-known UCI repository is used to evaluate the performance of the proposed CS-SVM model. The dataset is composed of the RSSI values of seven Wi-Fi access points collected from four different rooms. The variation in RSSI values of Wi-Fi access point dramatically decreases classification accuracy and effect value of other performance parameters. Furthermore, the formation of fingerprinting RSSI radio map is expensive in terms of labor and time. The proposed model is evaluated in terms of accuracy, precision, and recall with KNN and SVM using MATLAB. The proposed CS-SVM model achieves high accuracy of up to 99.7% as compared to KNN (98.7%) and SVM (98.3%).

Data Availability

The dataset is available at IndoorIndustrialLocalisationDataset (https://github.com/vauchey/IndoorInsdustrialLocalisationDataset/).

Conflicts of Interest

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

References

[1] W. Sakpere, M. Adeyeye-Oshin, and N. B. Mlitwa, “A state-of-the-art survey of indoor positioning and navigation systems and technologies,” *South African Computer Journal*, vol. 29, no. 3, pp. 145–197, 2017.

[2] W. Xue, K. Yu, X. Hua, Q. Li, W. Qiu, and B. Zhou, “APs’ virtual positions-based reference point clustering and physical distance-based weighting for indoor wi-fi positioning,” *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3031–3042, 2018.

[3] S. He and S. H. G. Chan, “Wi-Fi fingerprint-based indoor positioning: recent advances and comparisons,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 466–490, 2015.

[4] W. K. Zegeye, S. B. Amsalu, Y. Astatke, and F. Moazzami, “WiFi RSS fingerprinting indoor localization for mobile devices,” in *Proceedings of the 2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1–6, New York, NY, USA, October 2016.

[5] A. K. M. M. Hossain and W.-S. Soh, “A survey of calibration-free indoor positioning systems,” *Computer Communications*, vol. 66, pp. 1–13, 2015.

[6] S. S. Mohar, S. Goyal, and R. Kaur, “A survey of localization in wireless sensor network using optimization techniques,,” in *Proceedings of the 2018 4th IEEE International Conference On Computing Communication And Automation (ICCCA)*, pp. 1–6, Greater Noida, India, December 2018.

[7] K. Sabanci, E. Yigit, D. Ustun, A. Toktas, and M. F. Aslan, “Wifi based indoor localization: application and comparison of machine learning algorithms,,” in *Proceedings of the 2018 IEEE XXIIIrd International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory (DIPED)*, pp. 246–251, Tbilisi, Georgia, September 2018.

[8] J. Minlan, L. Jingyuan, and Z. Xiaokang, “Research on algorithm of three-dimensional wireless sensor networks node localization,” *Journal of Sensors*, vol. 2016, Article ID 2745109, 9 pages, 2016.

[9] X.-S. Yang and S. Deb, “Cuckoo search: recent advances and applications,” *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.

[10] M. A. E. Aziz and A. E. Hassanien, “Modified cuckoo search algorithm with rough sets for feature selection,” *Neural Computing and Applications*, vol. 29, no. 4, pp. 925–934, 2018.

[11] A. Yassin, Y. Nasser, M. Awad et al., “Recent advances in indoor localization: a survey on theoretical approaches and applications,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1327–1346, 2016.

[12] F. Subhan, S. Ahmed, and K. Ashraf, “Extended gradient predictor and filter for smoothing RSSI,” “, in *Proceedings of the 16th IEEE International Conference on Advanced Communication Technology*, pp. 1198–1202, Pyeong Chang, Feburary 2014.

[13] R. Bruha and P. Kriz, “Different approaches to indoor localization based on Bluetooth low energy beacons and wi-fi,” in *Proceedings of the International Conference on Computational Collective Intelligence*, pp. 305–314, Nicosia, Cyprus, September 2017.

[14] L. Pei, D. Liu, D. Zou, R. Lee Fook Choy, Y. Chen, and Z. He, “Optimal heading estimation based multidimensional particle filter for pedestrian indoor positioning,” *IEEE Access*, vol. 6, pp. 49705–49720, 2018.

[15] P. Sthapit, H. S. Gang, and J. Y. Pyun, “Bluetooth based indoor positioning using machine learning algorithms,,” in *Proceedingsa of the 2018 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pp. 206–212, Jeju, June 2018.

[16] A. Xiao, R. Chen, D. Li, Y. Chen, and D. Wu, “An indoor positioning system based on static objects in large indoor scenes by using smartphone cameras,” *Sensors*, vol. 18, no. 7, pp. 1–17, 2018.

[17] J. Y. Hsieh, C. H. Fan, J. Z. Liao, J. Y. Hsu, and H. Chen, “Study on the Application of Indoor Positioning Based on Low Power Bluetooth Device Combined with Kalman Filter and Machine Learning,” *EasyChair Preprint*, pp. 1–9, 2019.

[18] S. P. Rana, J. Prieto, M. Dey, S. Dudley, and J. M. Corchado, “A self regulating and crowdsourced indoor positioning system through wi-fi fingerprinting for multi storey building,” *Sensors*, vol. 18, no. 11, pp. 1–15, 2018.

[19] A. K. Taskan, “Precise Indoor Positioning Using Bluetooth Low Energy (BLE) Technology,” *Middle East Technical University*, vol. 21, no. 3, 971 pages, 2019.

[20] X. Jiang, Y. Chen, J. Liu, Y. Gu, and L. Hu, “FSELM: fusion semi-supervised extreme learning machine for indoor localization with Wi-Fi and Bluetooth fingerprints,” *Soft Computing*, vol. 22, no. 11, pp. 3621–3635, 2018.

[21] G. Li, E. Geng, Z. Ye, Y. Xu, and H. Zhu, “An indoor positioning algorithm based on RSSI real-time correction,”” in *Proceedings of the 2018 14th IEEE International Conference on Signal Processing (ICSP)*, pp. 129–133, Beijing, China, August 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5531185, 12 pages
https://doi.org/10.1155/2021/5531185

Research Article

Optimizing Quality of Service of Clustering Protocols in Large-Scale Wireless Sensor Networks with Mobile Data Collector and Machine Learning

Rahma Gantassi,[1] **Bechir Ben Gouissem**,[1] **Omar Cheikhrouhou**,[2] **Salim El Khediri**,[3,4] **and Salem Hasnaoui**[1]

[1]*Communication System Laboratory Sys'Com (ENIT), University of Tunis El Manar (UTM), Tunis, Tunisia*
[2]*College of Computers and Information Technology, Taif University, Taif 21944, Saudi Arabia*
[3]*Department of Information Technology, College of Computer, Qassim University, Buraydah, Saudi Arabia*
[4]*Department of Computer Sciences, Faculty of Sciences, University of Gafsa, Gafsa, Tunisia*

Correspondence should be addressed to Rahma Gantassi; rahmagantassii@gmail.com

Received 30 January 2021; Revised 22 February 2021; Accepted 4 March 2021; Published 15 March 2021

Academic Editor: Shafiq Muhammad

The rise of large-scale wireless sensor networks (LSWSNs), containing thousands of sensor nodes (SNs) that spread over large geographic areas, necessitates new Quality of Service (QoS) efficient data collection techniques. Data collection and transmission in LSWSNs are considered the most challenging issues. This study presents a new hybrid protocol called MDC-K that is a combination of the K-means machine learning clustering algorithm and mobile data collector (MDC) to improve the QoS criteria of clustering protocols for LSWSNs. It is based on a new routing model using the clustering approach for LSWSNs. These protocols have the capability to adopt methods that are appropriate for clustering and routing with the best value of QoS criteria. Specifically, the proposed protocol called MDC-K uses machine learning K-means clustering algorithm to reduce energy consumption in cluster head (CH) election phase and to improve the election of CH. In addition, a mobile data collector (MDC) is used as an intermediate between the CH and the base station (BS) to further enhance the QoS criteria of WSN, to minimize time delays during data collection, and to improve the transmission phase of clustering protocol. The obtained simulation results demonstrate that MDC-K improves the energy consumption and QoS metrics compared to LEACH, LEACH-K, MDC maximum residual energy leach, and TEEN protocols.

1. Introduction

Over the past few years, wireless technology has continued to grow through technological developments in various fields related to the microelectronics field. Simultaneously to WSN, new fields have emerged and challenges have been taken up to meet the needs of people and the demands of various application areas such as the environment, industry, and culture. These new applications have motivated researchers to sought solutions to achieve application requirements. Indeed, in spite of the outstanding achievements in this area, many problems still remain to be solved. For instance, some new protocols for the WSN have been proposed in order to handle media access control, routing protocols, data aggregation, and so on. Many research efforts have been carried out on sensor network energy consumption minimization, but only a few have focused on improving clustering protocol based on Quality of Service (QoS) criteria. As a result, clustering protocols have become the key target for sensor optimization studies.

The adoption of machine learning, such as the K-means algorithm, with the LEACH protocol [1] during cluster head (CH) election yields significant gains in network lifetime, throughput, CH number, latency, and power consumption

[1]. Nevertheless, multihop routing adopted by LEACH-K [1] can influence energy consumption, throughput, reliability, and latency time. Therefore, it is very important to address the problem of poor stability and latency that save energy and all QoS essentially in terms of throughput. On the other hand, mobile data collector (MDC) is used as an intermediate between the CH and the Base Station (BS) to further improve the QoS criteria of WSN, to minimize time delays during data collection, and to extend the lifespan of the network in WSN. Therefore, this study proposed a new protocol, which is a combination of the LEACH-K approach and MDC, to improve the QoS of clustering protocols. Specifically, the purpose of the proposed protocol is to prolong the lifetime of the network and enhance its QoS criteria. As a result, an enhancement of the clustering protocol is proposed, which relies on network division in clusters before the election of the CH according to the machine learning K-means algorithm. An efficient mobile data collection system is advocated as well based on data aggregation that operates as clusters on the basis of the WSN that efficiently uses mobile nodes for data collection. Specifically, the contribution of this document is as follows:

(i) We have proposed a new routing protocol called MDC-K. The proposed algorithm first applies the machine learning K-means algorithm to split the network into K clusters. After that, it determines the CH of each cluster which will be the closest node to the centroid of each K-means cluster. Assign each CH to the sensor node (SN) having the minimum distance to the CH. The CH can broadcast data to MDC when MDC visits the nearest CH. MDC completes its collection and finally delivers the data to the BS. The MDC uses the location of each CH to calculate the distance between it and all the CHs and moves to the CH having the minimum distance.

(ii) We compared the proposed protocol with existing solutions and proved its performance.

The remainder of the paper is arranged as follows. Following a brief introduction to the study proposed in Section 1, some related works are outlined in Section 2. Next, in Section 3, the proposed approach is explained. Section 4 gives several simulation scenarios with simulation parameters and evaluates the performance of the proposed approach. Section 5 summarizes the main contribution of this work and gives the perspectives.

2. Literature Study

Several studies have been conducted to develop MDC in clustering protocols [2–15] where they have focused on the maximization of WSN's lifespan, while other studies have been conducted in order to assess the impact of the K-means algorithm on the clustering performances in WSNs [1, 16–20]. Particularly, some studies have integrated MDC into clustering protocol, whereas others have applied the machine learning K-means algorithm in the clustering protocol. The objective of this section is to review and summarize the recent studies on both K-means and MDC approaches in routing protocols. Some studies used MDC in clustering protocol; others used MDC in Large-Scale Wireless Sensor Networks (LSWSNs), whereas the rest of the studies used MDC in WSN in general. Table 1 summarizes some literature works.

Besides, many studies used the machine learning K-means clustering algorithm in WSN. In particular, the majority used the K-means algorithm in clustering protocol. The K-means approach can make the cluster of nodes close to the centroid, which decreases the energy consumption and increases the throughput. Furthermore, the integration of MDC into the routing protocol can reduce the data collection times and extends network life in WSNs. Moreover, clustering protocols contain two phases, namely, the CH election phase and the data transmission phase. As previously discussed, some studies have integrated the MDC in clustering protocol, whereas others have applied the machine learning K-means algorithm in clustering protocol. To the best of our knowledge, this work presents the first attempt to combine both MDC and K-means algorithm to reduce the energy consumption in clustering protocol.

3. The MC-K Proposed Protocol

To the best of our knowledge, no study has used the combination of K-means and MDC in clustering protocol, which is the aim of this study. Specifically, the K-means algorithm was integrated into the election step and MDC in the transmission step of the clustering protocol in order to improve the QoS metrics.

During data collection, the MDC follows a greedy approach, each time selects the nearest CH, and moves to it to collect its data. Afterward, the MDC sends all collected data to the BS. Our proposed protocol contains two phases, the initialization phase and the transmission phase.

3.1. Initialization Phase

3.1. Initialization Phase. This phase starts with the application of the K-means algorithm [21, 22] to the WSN. Then, the SNs are divided into K groups.

The output of the K-means algorithm is k centroids. The BS selects the SN with the minimum distance to these centroids to be the CH. After CH assignment, an advertisement message containing the CH identifier is broadcast. Once receiving this message, SNs send a request message containing their coordinates to the CH to join the cluster. Each CH chooses its member nodes that are closest to it. Specifically, the CHs select the nodes belonging to their K-means cluster. Then, the BS knows the coordinates of the CHs. Suppose that the SNs denoted by ni are divided into k clusters {C1, C2, C3, C4, C. . . . . ., CK}. Assume ck to be the centroid of the cluster CK. Therefore,

$$C1 \cup C2 \cup \ldots \cup CK = \sum_{i=1}^{N} n, \qquad (1)$$

where N is the total number of nodes in the network.

The goal of K-means is to reduce the overall Euclidean distance between the CH and all the members of the cluster.

Table 1: Comparative study of the literature.

Study	Approach	Advantages	Challenges
[2]	The authors proposed distance energy evaluated (DEE) approach: (i) Introducing the stratified network topology arrangement that provides resiliency in multihop communication data to the base station	The simulation shows that protocol behavior is significantly enhanced for reliability, lifetime, and energy savings	(i) In this approach, the authors did not focus on the QoS improvement
[3]	The author approved the steering convention based mainly on the multichannel plan MDC at the lower station: (i) This allocates channel to MDCs instead of the single channel	(i) This protocol demonstrates meaningful changes from LEACH and a single application-specific network protocol for WSNs. (ii) The conventions for using sensor hub vitality and improving system lifetime and efficient information assembly due to less use of vitality in the middle of information transmission	(i) Despite the good finding of this study, it did not improve QoS performance
[4]	MEACBM protocol addresses heterogeneous clustering taking into account 3 SNs levels of multihop for intercluster communication and connectivity of SNs throughout the network area	Simulation results indicate the performance of MEACBM protocols compared to other contemporaneous routing protocols based on clusters regarding network lifetime, stability, throughput, channels, and dead nodes	(i) MEACBM protocol improves QoS performance
[5]	The proposed approach used MDC minimum distance leach which is based on the multihop communication between BS, SNs, and MDC	The MDC minimum distance leach approach outperforms the hybrid multihop LEACH approach regarding energy consumption and greatly improves lifetime	(i) Despite the good finding of this study, it did not improve QoS performance
[6]	The authors proposed analytic methods to compute node power consumption and identify the optimum node clusters to be used multihop and single-hop models	This approach can model large-scale networks (10,000 nodes and more) and test various network scenarios in little time. (i) For multihop, it demonstrated that the increase of clusters does not always reduce the energy consumption of the nodes owing to the DREQ problem flooding. (ii) For single hop, sensor nodes would have to spend a significant quantity of energy to send to the mobile access point when the number of clusters is lower in LSWSNs	(i) In this approach, the authors did not focus on the QoS improvement
[7]	The authors used the multilevel MDC method to assess the lifespan of the network in an open and dense agricultural area. Specifically, this study provided an effective method for extending the lifespan based on automaton learning at WSN.	(i) Experimental results showed a significant improvement in network lifetime extension compared to previous methodologies	(i) This approach did not ameliorate the QoS
[8]	The authors proposed MDC maximum residual energy protocol: (i) The authors presented a comparative study between MDC maximum residual energy routing protocol and LEACH multihop hybrid routing protocol	(i) Simulation results showed significantly better performance of the proposed maximum residual energy MDC protocol than the LEACH multihop hybrid routing protocol regarding the lifetime of the network	(i) The simulation parameters were very limited as in [14]. Except in [14], the MDC moves toward the nodes with the minimum distance appearing at the MDC, while in [8], MDC moves toward the nodes with maximum energy.
[9]	In the proposed approach: (i) The network is divided into equal clusters. (ii) In particular, [9] has integrated the fine-tuning of the node transmission range and the adjustment of the MDC speed	(i) Simulation findings showed significant reductions in the energy consumption of the CH	(i) In this approach, the authors did not focus on the QoS improvement. (ii) This approach did not ameliorate the QoS
[10]	In the proposed approach [10], the MDC begins its data collection journey from the data sink by getting a signal of the CH beacon, then collects the data directly at the CH, and conducts the data to the BS	(i) With this method, the lifetime of the sensors can be extended because the method is reactive	(i) In [10], the authors however did not make any simulation to validate the proposed approach

Table 1: Continued.

Study	Approach	Advantages	Challenges
[11]	In [11], MDC can interrogate one by one the sensors located nearby upon receipt of the acknowledgment; a sensor loads the data directly into the MDC and does not transmit the data back to another SN	(i) This approach has the potential to significantly extend the network lifetime relative to the schematic with a static data collector and greatly reduce the length of displacement relative to the coverage line protocol	(i) This approach did not ameliorate the QoS
[12, 13]	The authors focused on the design of an LWSN-based MDC, in which the mobile node tours each SN within the network, collects the data from the SNs, and transports the collected data to the BS [12, 13]	(i) For the multihop approach, researchers in [12] decrease the network distance while utilizing several well nodes. Thus, the problem of hot spots is prevented and the number of hops can also be reduced, resulting in a reduction of energy consumption	(i) This approach did not ameliorate the QoS
[14]	(i) The authors proposed, analyzed, and validated the MDC using the LEACH protocol in an environmental application	(i) This protocol shows considerable progress in the energy consumption of the sensor nodes. (ii) This protocol improves the life of the network	The simulation parameters were very limited (i) The maximal number of the used SNs was only 40 and the simulation area was equal to 1 Km^2
[15]	To avoid the premature death of sensors and avoid the problem of hot spots, a dynamic clustering approach is proposed in [15]. Furthermore, the technique of ant colony optimization (ACO) is used for the efficient selection of the mobile sinks path.	(i) The results of the simulation demonstrate that the CH election strategy based on the COA-based MDC greatly improves the WSN's lifespan. (ii) ACO was compared to current GA, PSO, and LEACH	(i) This approach did not ameliorate the QoS
[16]	(i) The author applied the K-means approach in WS	(i) The number of clusters is increasing. (ii) WSN's energy consumption is reduced	(i) The limitation to this approach is that a minimum number of parameters are used. For example, during the simulation phase, the authors focused only on 2 clusters with 10 nodes. (ii) This approach did not ameliorate the QoS.
[1]	In [1], the K-means algorithm was applied in LEACH protocol prior to the election of CH to reduce energy consumption and enhance the Quality of Service criteria	(i) The results showed that the proposed protocol decreased power consumption and latency and improved network stability, lifetime, and throughput. (ii) The mechanisms of aggregation and LEACH-K deteriorate the QoS leading to reduced stability and latency	(i) This approach did not ameliorate the QoS (ii) This approach did not apply the MDC (iii) This approach had not been tested in LSWSNs
[17]	The authors analyzed the routing protocol and started from the point of view of cluster routing in order to investigate the classical ACH protocol and evaluate its advantages and drawbacks	Based on this analysis, an aggregation process based on the K-means algorithm is provided and a verification function is appended to the K-means algorithm for dynamically adjusting the system to obtain stable system operation and transmitting information while minimizing the system's energy consumption	(i) This approach did not ameliorate the QoS (ii) This approach did not apply the MDC (iii) This approach had not been tested in LSWSNs
[18]	The authors analyzed the performance of KM and FCM and investigated which of them is best suited to build balanced clusters, allowing researchers to choose the right algorithm to improve network lifetime	Simulation results show that FCM is more superior to KM by producing balanced clusters with the random distribution manner for SN	(i) This approach did not ameliorate the QoS (ii) This approach did not apply the MDC (iii) This approach had not been tested in LSWSNs
[19, 20]	The authors of [19, 20] proposed two hybrid clustering algorithms, K-means particle swarm optimization (KPSO) and K-means genetic algorithms (KGAs), which show improvements compared to traditional LEACH in energy saving	The results showed that the proposed protocol reduced energy consumption	(i) This approach did not ameliorate the QoS (ii) This approach did not apply the MDC

It reduces the sum of the distances between SNs and the cluster centroid as shown in Figure 1.

The BS decides the nodes to be CH, based on two conditions: the closer node from the centroid and the total number of CHs should not be greater than the probability value.

$$P(t) = \frac{K}{N-K}. \tag{2}$$

So, if the energy of the nodes differs, the probability Pi (t) relates to the remaining energy at each node. This probability will therefore be equal to

$$Pi(t) = \frac{Ei(t)}{\mathrm{Etotal}(t)}K, \tag{3}$$

where $Ei(t)$ is the residual energy relating to each node ni. This probability favorises the nodes with more energy resource to become CHs. $\mathrm{Etotal}(t)$ is the total energy available in the network.

3.2. Data Transmission Phase. The MDC moves toward the nearest CH and collects its data. MDC maintains a set of already visited CHs and then goes to the nearest CH from the remaining CHs until it collects all data from all CHs. Afterward, it sends all the data to the BS. Afterward, it sends all the data to the BS. Figure 2 explains the flowchart of data transmission from CH to MDC.

4. Simulation Results and Analysis

In order to evaluate the performance of the proposed approach, it is compared with various existing protocols based on QoS criteria such as throughput and energy consumption in the WSN. The simulation of the proposed approach is done using MATLAB and various parameters and factors are taken into consideration while performing the simulation to enhance the network performance. In order to obtain these results, the value of K was set to 10 (10 is the optimal value of k for the best QoS criteria values according to [1]). Table 2 describes the simulation parameters common to all scenarios.

In these simulation scenarios, we compared the MDC-K protocol with LEACH, LEACH-K, TEEN, and MDC maximum residual energy protocol based on residual energy, throughput, and latency time of network in 10000 rounds. One hundred nodes have been chosen to test the performance of the MDC-K protocol. Figure 2 compares the residual energy of MDC-K against LEACH, LEACH-K, TEEN, and MDC maximum residual energy.

As indicated in the above figures, the proposed solution improves the network lifetime with higher performances. It improves the SN residual energy as well and numerical results show that the proposed method is able to decrease the SN energy consumption, which will result in more network lifetime. Figure 3 presents the residual energy and the throughput variations (when $K = 10$) of the proposed approach compared to LEACH, LEACH-K, TEEN, and MDC maximum residual energy leach. The graphs illustrated in Figure 3 depict that, by the integration of an unsupervised algorithm which is the K-means before the CH election and MDC in data transmission phase, total residual energy and throughput increase. The sensor network can be coated with an acceptable amount of energy (more than 0,027mj) in comparison to 0 of LEACH, 0 of TEEN, 0 of LEACH-K, and 0 of MDC maximum residual energy leach during 10000 rounds. Specifically, the MDC moves to its nearest CH to reduce energy consumption and consequently increases the residual energy.

As indicated in the above figures, the proposed solution improves the network lifetime with higher performances as the throughput is increased. Figure 4 presents the throughput variations (when $K = 10$) of the proposed approach compared

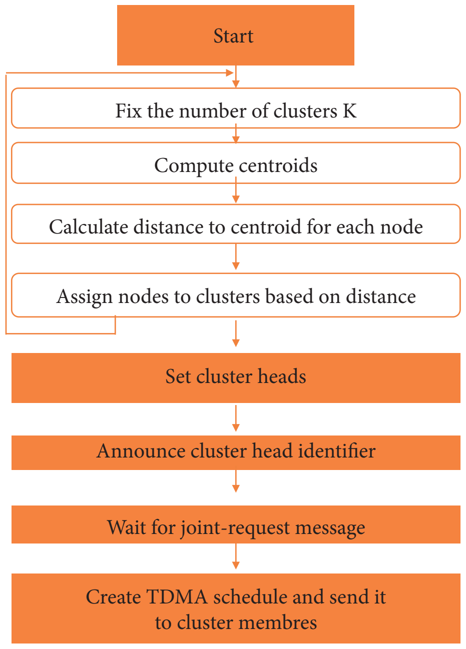

Start
Fix the number of clusters K
Compute centroids
Calculate distance to centroid for each node
Assign nodes to clusters based on distance
Set cluster heads
Announce cluster head identifier
Wait for joint-request message
Create TDMA schedule and send it to cluster membres

FIGURE 1: Initialization phase of the MDC-K.

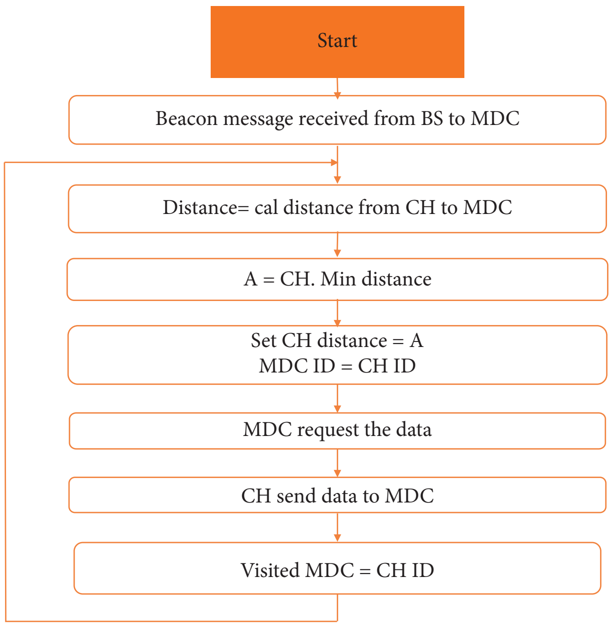


FIGURE 2: Data transmission from CH to MDC.

Table 2: Simulation parameters.

Settings	Values
Initial energy	Eelec = 0.5 J/node
Electronic dissipation energy (sending; receiving)	Eelec = 50 nJ/bit
Data aggregation energy	EDA = 5 nJ/bit/m^2
Transmit amplifier if dtoBS ≤ $d0$	Efs = 10pJ/bit/m^2
Transmit amplifier if dtoBS ≥ $d0$	Emp = 0.0013pJ/bit/m^4
Number of nodes	100nodes
LEACH routing percentage	0.05 (percentage of cluster head 5%)
Simulation area	(100 * 100)m

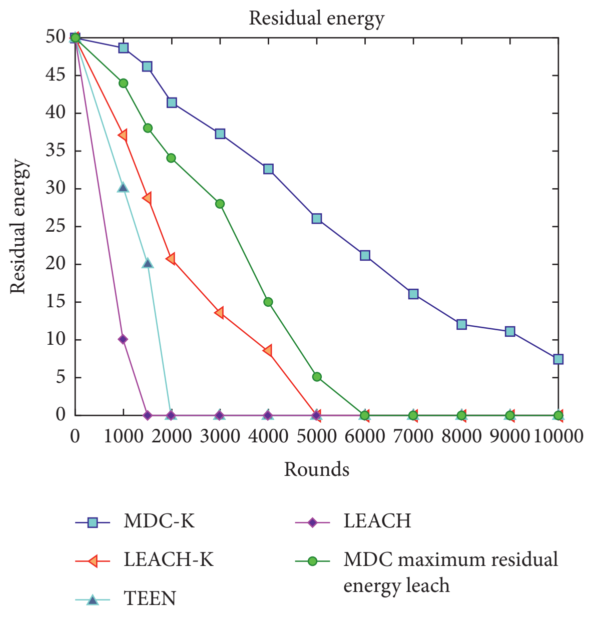


Figure 3: Comparison of residual energy between LEACH, LEACH-K, MDCEACH-K, TEEN, and MDC maximum residual energy leach protocols.

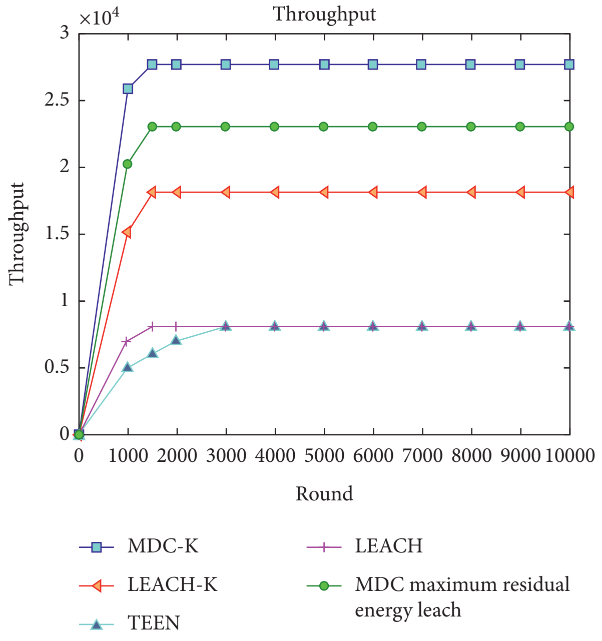


Figure 4: Comparison of throughput between LEACH, LEACH-K, MDCEACH-K, TEEN, and MDC maximum residual energy leach protocols.

to LEACH, LEACH-K, TEEN, and MDC maximum residual energy leach. The graphs illustrated in Figure 4 depict that, by the integration of an unsupervised algorithm which is the K-means before the CH election and MDC in data transmission phase, throughput increases.

Specifically, SNs can be coated with an acceptable amount of energy (more than 0,027J in comparison with 0 for LEACH, 0 for TEEN, 0 for LEACH-K, and 0 for MDC maximum residual energy leach during 10000 rounds) and can provide a considerable throughput (more than 37865 packets in comparison to 12009 for LEACH, 12001 for TEEN, 23300 for LEACH-K, and 17864 for MDC maximum residual energy leach during 10000 rounds). The simulation results are summarized in Table 3.

As shown in Table 3, the use of K-Means improved the election of the CH, whereas the use of MDC improved the routing. The effectiveness of the MDC-K protocol in LSWSNs was also tested. Specifically, we have evaluated the QoS criteria values in different area sizes. The latency time also falls down from 194,865 (ms) in LEACH, 153,74 (ms) in TEEN, 96,1602 (ms) in LEACH-K, and 51,551 in MDC maximum residual energy leach to 47,33 (ms) in MDC-K. Figure 5 presents a comparison between MDC-K, LEACH-K, TEEN, MDC maximum residual energy leach, and LEACH protocol in terms of stability, latency time, residual energy, and throughput in different area sizes.

Figure 5 showed that when the area size increases, the residual energy, throughput, and stability decreased slightly which means that the benefits of our protocol remain almost stable with a large area. For instance, the latency time also increases from 47,33 (ms) in area size (100, 100) to 50,505 (m) in area size (1000, 1000). The throughput decreases down from 37865 (packet) in area size (100, 100) to 37601 (packet) in area size (1000, 1000). According to its results, we conclude that our MDC-K is the best solution for LSWSNs. The stability decreases down from 2567 (rounds) in area size (100, 100) to 2407 (rounds) in area size (1000, 1000).

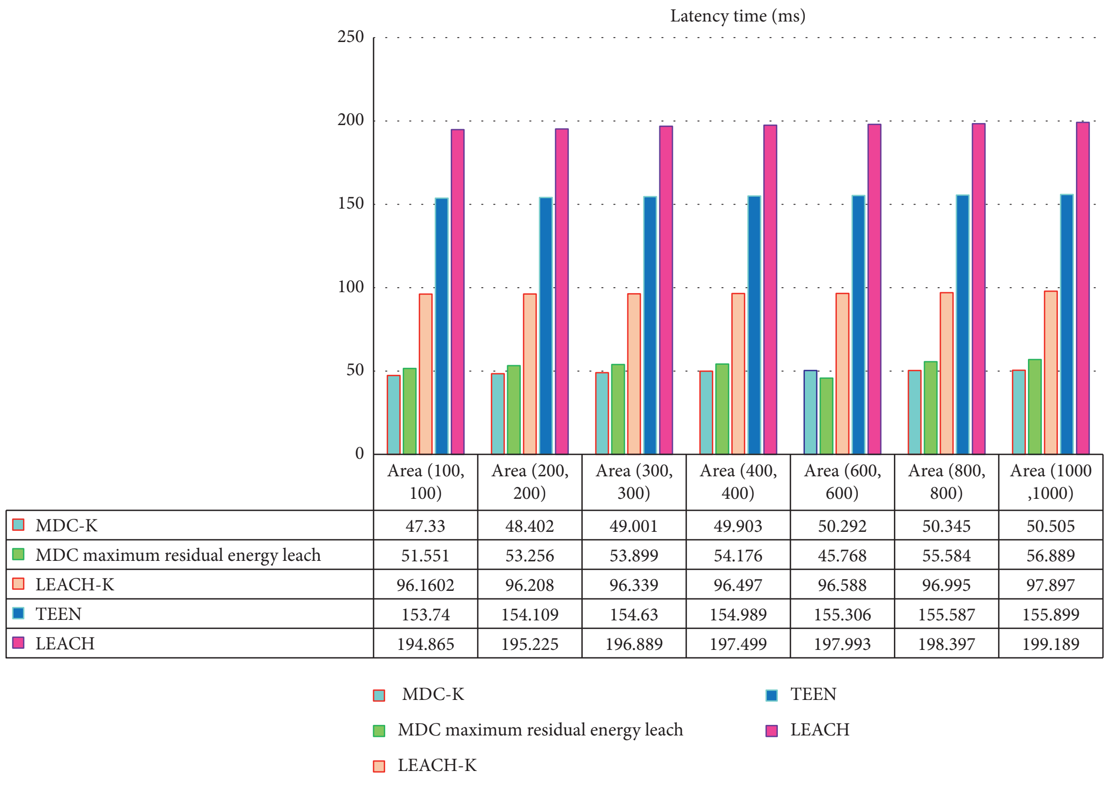
We have also tested the effect of the speed of MDC on QoS criteria. Figure 6 summarizes the simulation results of MDC-K with K-means by varying the speed of MDC.

We have also tested the effect of the speed of MDC on QoS criteria. Figure 6 summarizes the simulation results of MDC-K with K-means by varying the speed of MDC.

As shown in Figure 6, when the speed of MDC increases, the residual energy, throughput, and stability increases. This can

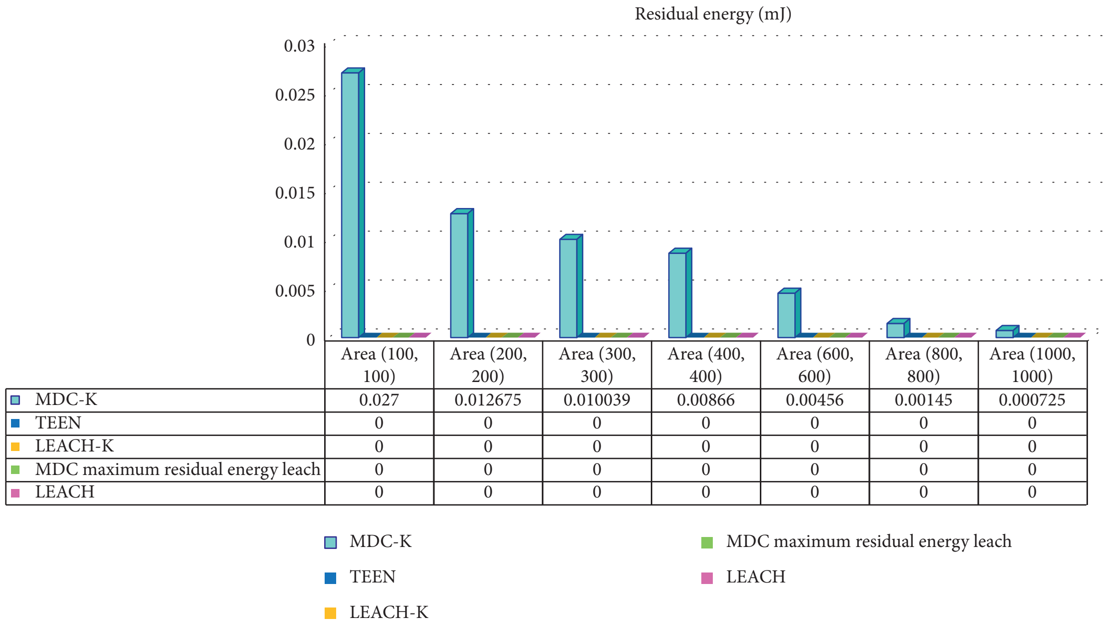
Table 3: Comparison of the network lifetime (residual energy), the stability, throughput, latency time, and number of CHs between the LEACH, TENN, LEACH-K, and MDC-K of 10000 rounds.

	LEACH	TEEN	LEACH-K	MDC maximum residual energy leach	MDC-K
Throughput (packet)	12009	12001	30017	23300	37865
Residuals energy (mJ)	0	0	0	0	≥0,027
Latency time (ms)	194,86	153,74	96,160	51,551	47,33
Stability (rounds)	799	1078	1399	1913	2567



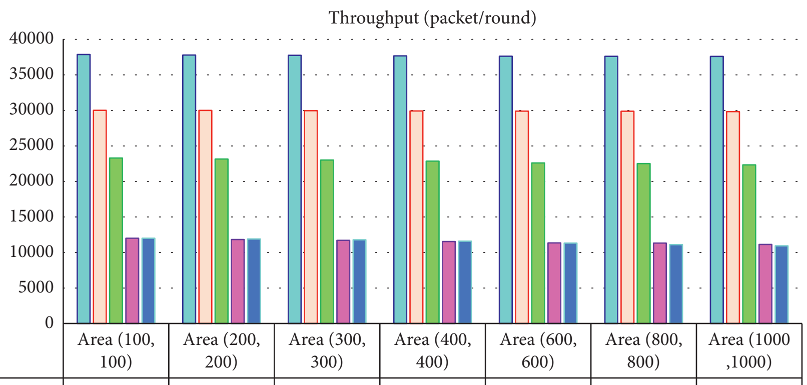
	Area (100, 100)	Area (200, 200)	Area (300, 300)	Area (400, 400)	Area (600, 600)	Area (800, 800)	Area (1000 ,1000)
MDC-K	47.33	48.402	49.001	49.903	50.292	50.345	50.505
MDC maximum residual energy leach	51.551	53.256	53.899	54.176	45.768	55.584	56.889
LEACH-K	96.1602	96.208	96.339	96.497	96.588	96.995	97.897
TEEN	153.74	154.109	154.63	154.989	155.306	155.587	155.899
LEACH	194.865	195.225	196.889	197.499	197.993	198.397	199.189

(a)



	Area (100, 100)	Area (200, 200)	Area (300, 300)	Area (400, 400)	Area (600, 600)	Area (800, 800)	Area (1000, 1000)
MDC-K	0.027	0.012675	0.010039	0.00866	0.00456	0.00145	0.000725
TEEN	0	0	0	0	0	0	0
LEACH-K	0	0	0	0	0	0	0
MDC maximum residual energy leach	0	0	0	0	0	0	0
LEACH	0	0	0	0	0	0	0

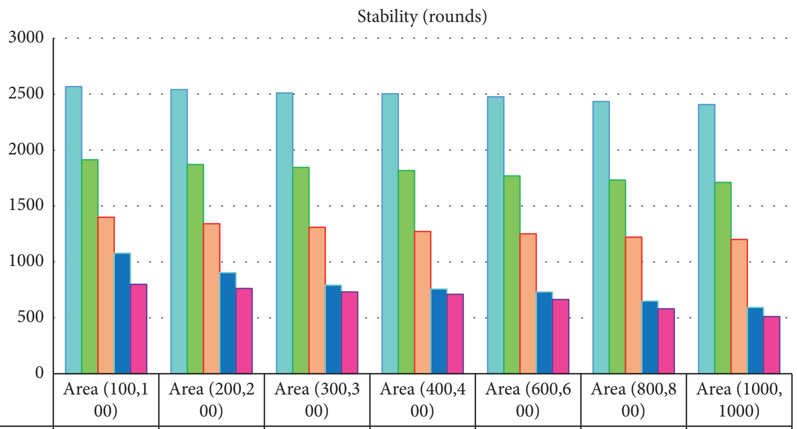
(b)



	Area (100, 100)	Area (200, 200)	Area (300, 300)	Area (400, 400)	Area (600, 600)	Area (800, 800)	Area (1000 ,1000)
🟦 MDC-LEACH-K	37865	37797	37750	37676	37632	37617	37601
🟥 LEACH-K	30017	29998	29964	29924	29901	29878	29838
🟩 MDC maximum residual energy leach	23300	23150	23009	22856	22603	22514	22327
🟪 LEACH	12009	11827	11711	11539	11344	11319	11131
🟦 TEEN	12001	11904	11792	11588	11322	11101	10933

🟦 MDC-LEACH-K 🟪 LEACH
🟥 LEACH-K 🟦 TEEN
🟩 MDC maximum residual energy leach

(c)



	Area (100,1 00)	Area (200,2 00)	Area (300,3 00)	Area (400,4 00)	Area (600,6 00)	Area (800,8 00)	Area (1000, 1000)
🟦 MDC-K	2567	2541	2511	2503	2476	2434	2407
🟩 MDC maximum residual energy leach	1913	1871	1845	1817	1769	1732	1711
🟧 LEACH-K	1399	1341	1309	1272	1251	1221	1201
🟦 TEEN	1078	904	792	758	732	650	593
🟪 LEACH	799	763	732	710	664	581	511

🟦 MDC-K 🟦 TEEN
🟩 MDC maximum residual energy leach 🟪 LEACH
🟧 LEACH-K

(d)

FIGURE 5: Comparison between MDC-K, LEACH-K, TEEN, MDC maximum residual energy leach, and LEACH protocol in terms of stability, latency time, residual energy, and throughput in different area sizes.

show the importance of MDC in routing protocols. In order to evaluate the effect of the density of nodes on the improvement of the QoS. We have simulated MDC-K with a different number of nodes. Figure 7 presents the results of the simulation.

Results showed that the variation of density of nodes keeps the same benefit and improvement with residual energy, stability, throughput, and latency time. As shown in Figure 7, when the number of nodes increases, the latency slightly increases, but it is still less than the latency of LEACH and LEACH-K protocols.

5. Discussion

To evaluate the efficiency of the proposed protocol in enhancing QoS of clustering protocol, Table 4 compares this protocol with similar protocols reported in the literature.

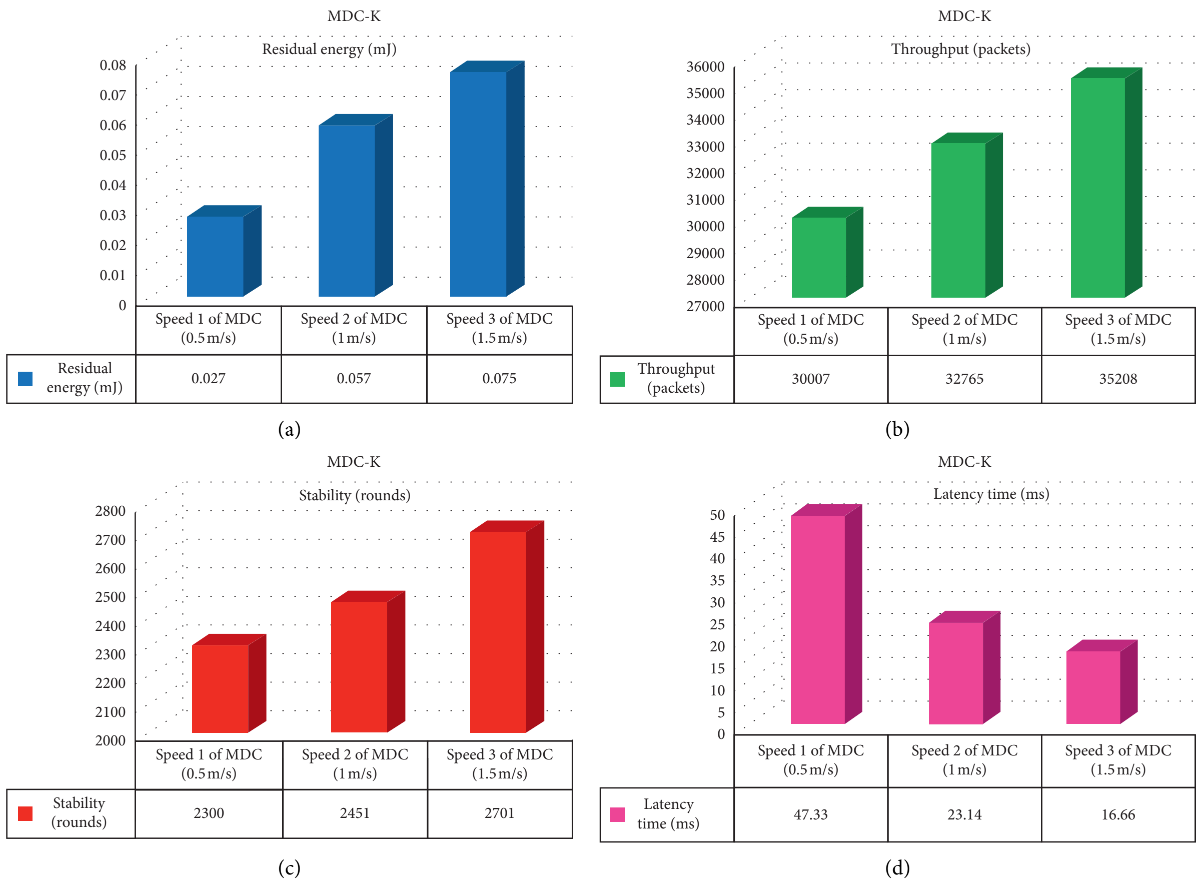


Figure 6: Comparison of the network lifetime (residual energy), the stability, the throughput, latency time of MDC-K with various MDC velocities during 10000 rounds.

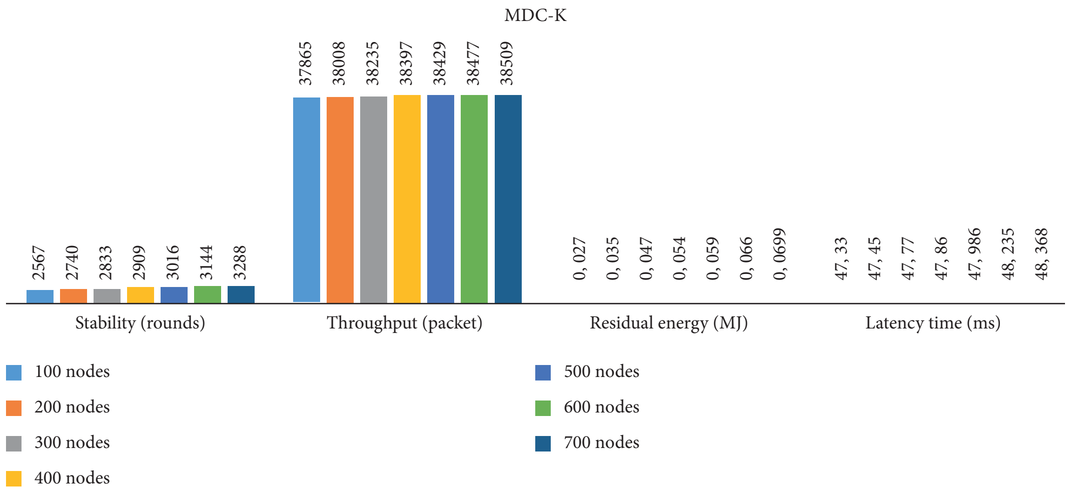


Figure 7: Comparison of the network lifetime (residual energy), stability, throughput, and latency time of MDC-K with a various number of nodes.

As shown in Table 4, several protocols were proposed for enhancing the LEACH protocol. For instance, the LEACH-K protocol is proposed for improving all the metrics of QoS. MDC-K approach also improved residual energy, latency, stability, and throughput. However, LEACH-K was not useful for LSWSNs. The MDC-K protocol is better than MDC minimum distance LEACH, MDC maximum residual energy leach, and DV-MDC because it improves all the QoS criteria. For example, MDC-K has 37865 (packet) throughput value more than 23300 (packet) in MDC maximum residual energy leach, 23422 (packet) in MDC maximum residual energy leach, and 714 (packet) in DV-

Table 4: Comparison of the network lifetime, stability, throughput, and latency time between MDC-K and similar protocols reported in the literature.

	Type of network	Energy efficiency	Stability	CH election	Throughput	Network lifetime	Latency time
LEACH-K [1]	Homogeneous WSN	Higher than LEACH and TEEN	More stable than TEEN and LEACH	Based on initial and residual energy	More than TEEN, LEACH, MDC minimum distance leach, and MDC maximum residual energy leach	Higher than TEEN and LEACH	Lesser than LEACH and TEEN
MDC-K (this paper)	Homogeneous WSN	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, energy-efficient cluster head LEACH, DV-MDC, and MEACBM	More stable than TEEN, LEACH, LEACH-K, MDC minimum distance leach, and MDC maximum residual energy leach	Based on residual energy	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, energy-efficient cluster head LEACH, MEACBM, and DV-MDC	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, energy-efficient cluster head LEACH, and DV-MDC	Lesser than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, MEACBM, and energy-efficient cluster head LEACH
MDC minimum distance leach [5]	Homogeneous WSN	Higher than TEEN, LEACH, and LEACH-K	More stable than LEACH, TEEN, and LEACH-K	Based on initial and residual energy	More than TEEN and LEACH	Higher than TEEN, LEACH, and LEACH-K	Lesser than TEEN, LEACH, and LEACH-K
MDC maximum residual energy leach [8]	Homogeneous WSN	Higher than TEEN, LEACH, and LEACH-K	More stable than LEACH, TEEN, and LEACH-K	Based on initial and residual energy	More than TEEN and LEACH	Higher than TEEN, LEACH, and LEACH-K	Lesser than TEEN, LEACH, LEACH-K, and MDC minimum distance leach
DV-MDC [23]	Homogeneous WSN	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, and energy-efficient cluster head LEACH	More stable than LEACH, TEEN, MDC minimum distance leach, MDC maximum residual energy leach, and energy-efficient cluster head LEACH	Based on initial and residual energy	More than TEEN, LEACH, energy-efficient cluster head LEACH, MDC minimum distance leach, and MDC maximum residual energy leach	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, and energy-efficient cluster head LEACH	Lesser than LEACH, LEACH-K, MDC minimum distance leach, MDC maximum residual energy leach, and energy-efficient cluster head LEACH
Energy-efficient cluster head LEACH [14]	Homogeneous WSN	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, and MDC maximum residual energy leach	More stable than LEACH, TEEN, MDC minimum distance leach, MDC maximum residual energy leach, and energy-efficient cluster head LEACH	Based on initial and residual energy	More than LEACH and TEEN	Higher than TEEN, LEACH, LEACH-K, MDC minimum distance leach, and MDC	Lesser than TEEN and LEACH

Table 4: Continued.

	Type of network	Energy efficiency	Stability	CH election	Throughput	Network lifetime	Latency time
MDC/PEQ [28]	Homogeneous WSN	Higher than TEEN, LEACH, and LEACH-K	More stable than LEACH and TEEN	Based on residual energy	More than TEEN, LEACH, energy-efficient cluster head LEACH, MDC minimum distance leach, and MDC maximum residual energy leach	Higher than TEEN, LEACH, and LEACH-K	Lesser than TEEN and LEACH, LEACH-K, and energy-efficient cluster head LEACH
MEACBM [14]	Homogeneous WSN	Higher than TEEN, LEACH, energy-efficient cluster head LEACH, MDC minimum distance leach, MDC maximum residual energy leach, and LEACH-K	More stable than TEEN, LEACH, energy-efficient cluster head LEACH, MDC minimum distance leach, and MDC maximum residual energy leach	Based on residual energy	More than TEEN, LEACH, energy-efficient cluster head LEACH, MDC minimum distance leach, and MDC maximum residual energy leach	Higher than TEEN, LEACH, MDC minimum distance leach, MDC maximum residual energy leach, and LEACH-K	Lesser than TEEN, LEACH, LEACH-K, and energy-efficient cluster head LEACH
DEE [2]	Homogeneous WSN	Higher than LEACH, optimum LEACH, TEEN, MDC minimum distance leach	More stable than LEACH, optimum LEACH, TEEN, and MDC minimum distance leach	Based on residual energy and distance to base station	More than LEACH, optimum LEACH, TEEN, and MDC minimum distance leach	Higher than LEACH, optimum LEACH, TEEN, and MDC minimum distance leach	Lesser than LEACH, optimum LEACH, TEEN, and MW-LEACH

MDC. The combination of the K-means and MDC approach improves all the QoS criteria of the LEACH-K protocol.

6. Conclusion

In this paper, we have proposed a new hybrid protocol called MDC-K, which is a combination of LEACH-K approach and MDC. Specifically, this protocol uses the K-means algorithm to reduce energy consumption in the CH election phase. In addition, MDC is used as an intermediate between the CH and the BS to further enhance the QoS criteria of WSN, to minimize time delays during data collection, and to extend the lifetime of the network in WSN. The obtained simulation results demonstrate that MDC-K has a considerable impact on energy consumption and QoS metrics. Particularly, this protocol achieves a significant improvement in terms of energy consumption, residual energy, throughput, latency time, and stability gains better than LEACH, TEEN, LEACH-K, and maximum residual energy LEACH protocols. Moreover, the simulation results show that MDC-K is more suited for LSWSNs as it maintains a good performance in energy consumption and all studied QoS criteria for the large supervised area.

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

Dr. Omar Cheikhrouhou acknowledges Taif University for supporting this study through Taif University Researchers Supporting Project no. TURSP-2020/55, Taif University, Taif, Saudi Arabia.

References

[1] R. Gantassi, B. Ben Gouissem, and J. Ben Othmen, "Routing protocol LEACH-K using K-means algorithm in wireless sensor network," in *Springer Nature Switzerland AG 2020 L*, Barolli, Ed., pp. 299–309, WAINA 2020, AISC, Caserta, Italy, 2020.

[2] S. E. Khediri, N. Nasri, R. U. Khan, and A. Kachouri, "An improved energy efficient clustering protocol for increasing the life time of wireless sensor networks," *Wireless Personal Communications*, vol. 116, no. 1, pp. 539–558, 2020.

[3] M. Tech and S. Bhatnagar, "Energy preserve using LEACH protocol in wireless sensor networks," *International Journal of Scientific Research & Engineering Trends*, vol. 5, no. 6, 2019.

[4] A. S. Toor and A. K. Jain, "Energy aware cluster based multi-hop energy efficient routing protocol using multiple mobile nodes (MEACBM) in wireless sensor networks," *AEU-*

International Journal of Electronics and Communications, vol. 102, pp. 41–53, 2019.

[5] R. Vishnuvarthan, R. Sakthivel, V. Bhanumathi, and K. Muralitharan, "Energy-efficient data collection in strip-based wireless sensor networks with optimal speed mobile data collectors," *Computer Networks*, vol. 156, pp. 33–40, 2019.

[6] K. Li and M. Ang, "Optimizing energy consumption for big data collection in large-scale wireless sensor networks with mobile collectors," *IEEE Systems Journal*, vol. 12, no. 1, 2018.

[7] M. Kamarei, A. Patooghy, M. Z. Shahsavari, and M. Javad Salehi, "Lifetime expansion in WSNs using mobile data collector: a learning automata approach," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 1, pp. 65–72, 2020.

[8] M. Arshad, Y. A. Mohamed, and A. S. Farhan, "Energy efficient cluster based routing scheme for mobile wireless sensor networks," in *Proceedings of the 5th International Conference on Intelligent and Advanced Systems (ICIAS)*, Kuala Lumpur, Malaysia, June 2014.

[9] S. K. Singh, P. Kumar, and J. P. Singh, "A survey on successors of leach protocol," *IEEE Access*, vol. 5, pp. 4298–4328, 2017.

[10] B. Mrinal Kanti Deb and D. Sudeshna, "Data gathering mechanism of mobile data collector in wireless sensor network," in *Proceedings of the 2016 International Conference on Internet of Things and Applications (IOTA)*, Maharashtra Institute of Technology, Pune, India, January 2016.

[11] M. Sajid, K. Khan, U. Qasim, Z. a. Khan, S. Tariq, and N. Javaid, "A new linear cluster handling (LCH) technique toward's energy efficiency in linear WSNs," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, vol. 7, pp. 389–393, Gwangju, Korea, March 2015.

[12] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "A framework for using unmanned aerial vehicles for data collection in linear wireless sensor networks," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1-2, pp. 437–453, 2013.

[13] I. Jawhar, M. Ammar, S. Zhang, J. Wu, and N. Mohamed, "Ferry-based linear wireless sensor networks," in *Proceedings of the GLOBECOM-IEEE Global Communications Conference*, pp. 304–309, Atlanta, GA, USA, October 2013.

[14] M. Arshad, N. Armi, N. Kamel, and N. M. SaadM, "Mobile data collector based routing protocol for wireless sensor networks," *Scientific Research and Essays*, vol. 6, no. 29, pp. 6162–6175, 2011.

[15] M. Krishnan, S. Yun, and M. J. Yoon, "Dynamic clustering approach with ACO-based mobile sink for data collection in WSNs," *The Journal of Mobile Communication, Computation and Information*, vol. 25, pp. 4859–4871, 2018.

[16] G. Devi, "The K-means clustering used in wireless sensor network," *International Journal on Computer Science and Engineering (IJCSE) ISSN*, vol. 8, no. 4, pp. 975–3397, 2016.

[17] E. Rabiaaa, B. Nourab, and C. Adnenec, "Improvements in LEACH based on K-means and Gauss algorithms," in *Proceedings of the International Conference on Advanced Wireless, Information, and Communication Technologies AWICT*, Tunis, Tunisia, October 2015.

[18] S. Randhawa and S. Jain Computer, "Performance analysis of LEACH with machine learning algorithms in wireless sensor network," *International Journal of Computer Applications*, vol. 147, no. 2, pp. 975–8887, 2016.

[19] Y. Feng, S. Zhao, and H. Liu, "Analysis of network coverage optimization based on feedback K-means clustering and artificial fish swarm algorithm," *IEEE Access Date of Current Version*, vol. 8, pp. 42864–42876, 2020.

[20] S. ElKhediria, W. Fakhetb, T. Moulahid, R. N. Khand, A. Thaljaouie, and K. Abdennaceur, "Improved node localization using K-means clustering for wireless sensor networks," *Computer Science Review*, vol. 37, Article ID 100284, 2020.

[21] B. Zhu, E. Bedeer, Ha H. Nguyen, R. Barton, and H. Jerome, "Improved soft-K-means clustering algorithm for balancing energy consumption in wireless sensor networks," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4868–4881, 2020.

[22] A. A.-H. Hassan, W. Md Shah, M. F. I. Othman, and H. A. H. Hassan, "Evaluate the performance of K-means and the fuzzy C-Means algorithms to formation balanced clusters in wireless sensor networks," *International Journal of Electrical & Computer Engineering*, vol. 10, no. 2, pp. 2088–8708, 2020.

[23] D. Kim, R. N. Uma, B. H. Abay, W. Wu, W. Wang, and A. O. Tokuta, "Minimum latency multiple data MULE trajectory planning in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 4, pp. 838–851, 2014.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5537041, 11 pages
https://doi.org/10.1155/2021/5537041

Research Article

An IoT Time Series Data Security Model for Adversarial Attack Based on Thermometer Encoding

Zhongguo Yang,[1] Irshad Ahmed Abbasi,[2] Fahad Algarni,[3] Sikandar Ali,[4,5] and Mingzhu Zhang[1]

[1]*School of Information Science and Technology, North China University of Technology, Beijing, China*
[2]*Department of Computer Science, Faculty of Science and Arts at Belgarn, University of Bisha, Sabt Al-Alaya 61985, Saudi Arabia*
[3]*College of Computing and Information Technology, Faculty of Computing and Information Technology, University of Bisha, Bisha 61922, Saudi Arabia*
[4]*Department of Computer Science & Technology, China University of Petroleum, Beijing 102249, China*
[5]*Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum-Beijing, Beijing 102249, China*

Correspondence should be addressed to Sikandar Ali; sikandar@cup.edu.cn

Received 21 January 2021; Revised 7 February 2021; Accepted 25 February 2021; Published 10 March 2021

Academic Editor: Shafiq Muhammad

Nowadays, an Internet of Things (IoT) device consists of algorithms, datasets, and models. Due to good performance of deep learning methods, many devices integrated well-trained models in them. IoT empowers users to communicate and control physical devices to achieve vital information. However, these models are vulnerable to adversarial attacks, which largely bring potential risks to the normal application of deep learning methods. For instance, very little changes even one point in the IoT time-series data could lead to unreliable or wrong decisions. Moreover, these changes could be deliberately generated by following an adversarial attack strategy. We propose a robust IoT data classification model based on an encode-decode joint training model. Furthermore, thermometer encoding is taken as a nonlinear transformation to the original training examples that are used to reconstruct original time series examples through the encode-decode model. The trained ResNet model based on reconstruction examples is more robust to the adversarial attack. Experiments show that the trained model can successfully resist to fast gradient sign method attack to some extent and improve the security of the time series data classification model.

1. Introduction

IoT amalgamates well-known products with state of the art infrastructures including distributed data storage, big data solutions, artificial intelligence (AI) utilities, or cloud [1]. Internet of Things (IoT) envisions connected, pervasive, and smart nodes link independently while providing all kinds of services. IoT data are collected at large to aid in decision-making. The IoT consumer products are no longer just the product only; it is the data, the product, the infrastructure, and the algorithms. These IoT products have switched to connected technologies from analog one, therefore, introducing novel risks for consumers regarding potential safety, privacy, and security issues for discriminatory data [2–4].

Moreover, Papernot et al. [5] have found that the adversarial samples are more transferable amongst various machine learning approaches, i.e., support vector machine, logistic regression, decision tree, and deep neural networks.

There are many application scenarios for IoT, as shown in Figure 1, including medical health, electricity, and intelligence device. There are also areas, which are very sensitive to attacks, such as industrial control decision support systems.

In other fields, such as State Grid and Industrial Control, the deep learning model built for them is prone to make decision errors due to data noise and deliberate attacks to modify data. For example, smart grids time series data were analyzed for electricity fraud detection, wherein these use cases perturbed data can succor thieves from being detected.

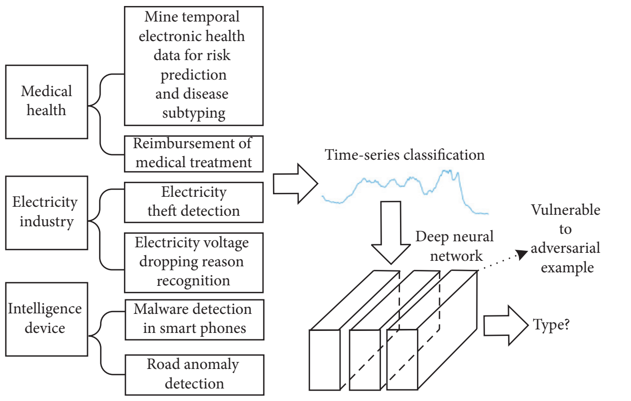

Medical health
Mine temporal electronic health data for risk prediction and disease subtyping
Reimbursement of medical treatment
Electricity industry
Electricity theft detection
Electricity voltage dropping reason recognition
Intelligence device
Malware detection in smart phones
Road anomaly detection
Time-series classification
Deep neural network
Vulnerable to adversarial example
Type?

Figure 1: Typical application fields of the deep neural network for IoT data.

As illustrated in Figure 1, in some sensitive and crucial systems, time-series data classification models are admired for their vast application. Thus, security and precisely the ability to detect the nodes being compromised, along with collecting and preserving evidence of malicious activities or an attack transpire as a priority in the triumphant deployment of IoT networks. Among these potential risks, AI algorithm security is rarely gained research interest although it is a very hot topic in the domain of AI.

Modern approaches in time-series data classification are based on the deep learning paradigm [6], specifically adversarial examples, which could lead to big recognition errors by adding small perturbation to the original time series.

The reason lies in the high dimension linear design of deep learning models. In order to better combat against the adversarial attack, we applied an encode-decode model to reconstruct time series examples from thermometer encoding of original time series. Although the classification models are not trained on the reconstruction examples, their training and valuation accuracy is the same as the model trained on the original examples. Moreover, we found that the new model is robust to the fast gradient sign method (FGSM) attack to some extent. To summarize, the contributions of this article are three folds as follows:

(1) Summarize some potential risks in the IoT time series classification (TSC) model

(2) Analyze the classification activation map and the attack area in time series

(3) A robust model-based encode-decode and thermometer encoding

The remaining paper is structured as follows. In Section two, we overview TSC works based on deep learning as well as attacks and defense methods in the fields of computer vision. Section three shows some potential risks in IoTTSC from different views. Section four presents a detailed description of our method and some basic theory. In the experiment section, we introduce the datasets, classification model architecture, and attack defense results. Finally, we analyze the defense effectiveness and give our future research directions.

2. Related Works

TSC problems are experienced in numerous real-life data mining tasks ranging from power consumption monitoring [4], food safety [7], and health care [2, 8, 9].

Deep learning has resolved some problems like pattern recognition in temporal and spatial data with higher accuracy that was thought to be impossible a few years ago. Fortunately, TSC tasks can be efficiently framed as deep learning problems; therefore, many researchers have recently begun to adopt deep learning models for TSC tasks [6].

The classification of time series IoT data is a key problem in various application domains. Backing the development of deep learning, investigators have started to work on the vulnerability of deep neural networks to adversarial attacks [10]. In the field of image processing, an adversarial attack alters original images in such a way that the modifications are nearly imperceptible by a human. The altered image is termed as an adversarial image, that will be confused by the neural network and will be misclassified, while that of the original image will be correctly classified. The well-known real-world attack includes modifying a traffic sign image so that it is misconceived by an autonomous vehicle [11]. Alteration of illegal content to make it undetectable by automatic moderation algorithms is another example. The most notable attack is gradient-based attack, where the attacker alters the image in the direction of the gradient of the loss function with reference to the input image and therefore escalates the rate of misclassification [12, 13].

The model of deep learning applied in a real environment on IoT data is fragile which is vulnerable to adversarial attack, and this has become a common problem of deep learning in other areas. At the same time, there are much security works for image processing such as defensive distillation [14], data compression [15], depth compression network [16], data randomization [17], and gradient regularization [18].

There are hardly any comprehensive studies on defense against an attack on temporal data. Fawaz et al. [8] discussed some serious problems in the classification of time-series data using a deep learning model. Different from the image, IoT time-series data own its special characteristics, such as dynamic changing and different sampling scale. Based on the characteristics of IoT data, this paper uses an encode-decode model-based deep neural network.

In the encode-decode stage, we used a thermometer coding method to be the decoded output. The reason to use the thermometer coding is to consider bringing a strong nonlinear transformation to the model. This is inspired by Goodfellow, who showed us the high dimension of the well-structured deep learning model. Buckman's et al. [19] work confirmed that the input discretization approach could repel against adversarial attacks. Inspired by these thoughts, different from the aforementioned works, we try to construct a whole network. In this network, the input is the original curves, and it will learn its original curve through the encode-decode model with its thermometer encoding as input. With the thermometer coding as input to the ResNet to predict its type, we will show the details of the proposed network and its effectiveness in the following parts.

3. Adversarial Attack in IoT Time Series Data

In this paper, we used Coffee's dataset [20] as typical time series data to illustrate the adversarial attack phenomena in IoT fields and ResNet [21] as a measure for neural network architecture.

3.1. Fast Gradient Sign Method and TSC Adversarial Attacks. Some adversarial examples and definition of the TSC problem were introduced by Fawaz et al. [8]. According to them, time series data can be mathematically represented as set $X = [x_1, x_2, \ldots, x_T]$. Let T is a real number and represents the length of X. Further, there is a well-trained deep learning model $f(\cdot) \in F\colon \mathscr{R}^T \longrightarrow \widehat{Y}$. Here, Y is the label space of time series, and $\mathscr{R}$ is a real number space. The adversarial example has to find another example X' to be a perturbed cloned version of X with the restriction that $X - X' < \varepsilon$ and $Y \neq Y'$. A visual illustration of given definitions is visualized in Figure 2.

The most classic adversarial method is the fast gradient sign method (FGSM). FGSM was first introduced by Goodfellow et al. [12] for generating adversarial images that trick the well-known GoogLeNet model. The attack is set up through a one-step gradient update in the direction of the gradient's sign at every single timestamp.

The perturbation procedure shown in Figure 3 can be represented mathematically as follows:

$$\eta = \varepsilon \cdot sign\left(\nabla_x \mathscr{J}\left(X, Y_{true}\right)\right), \tag{1}$$

where ε symbolizes the magnitude of the perturbation. The adversarial time series X' can be computed using $X' = X + \eta$.

Author of the FGSM paper mentioned the underlying reason why FGSM attacks the neural network. Firstly, the influence of disturbance in the neural network will be as big as snowball due to the linear design of the model. At present, ReLU is a kind of linear activation function in neural networks, which makes the whole network tend to be linear. Furthermore, the larger the dimension of input, the more vulnerable will be the model to adversarial attack.

3.2. The Distribution of Data in Adversarial Attack. Multidimensional scaling (MDS) [22] provides a possibility to get insights into the spatial distribution of the input time series. MDS project N-dimensional space into two-dimensional space while keeping the relative distance for any two time series. Given the nearest neighbor classifier achieving low accuracy on the raw time series, Euclidean distance (ED) could not be used directly in the raw data.

However, the high feature learned by the network could be used as a good presentation of the raw time series. Commonly, the perfectly connected layers in the last several layers of the neural network are often used as latent space, where the class-specific region differs for different classes.

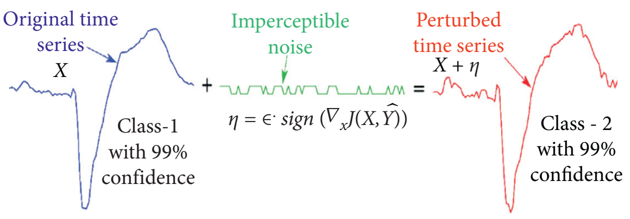


FIGURE 2: Adversarial examples taken from [8].

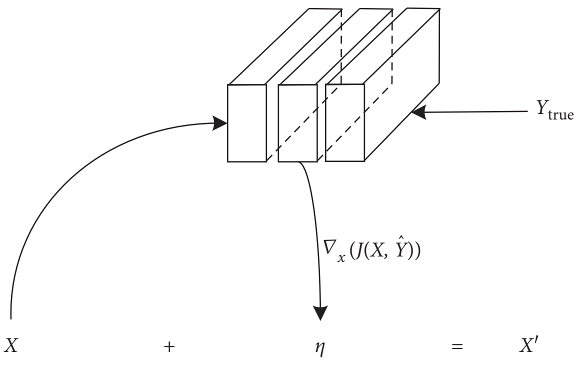


FIGURE 3: The principle of the fast gradient sign method.

We apply this method on ResNet, which achieves the best accuracy on most of the TSC problems [6]. In the ResNet architecture, there is a global average pooling (GAP) layer preceding the classifier layer. The GAP layer is a learned good representation of the raw time series, which is used to compute ED. When we get the distance for each pair of two time series, the metric MDS is a cost function called stress and can be obtained as follows:

$$\text{Stress}_D\left(X_1, \ldots, X_N\right) = \left(\frac{\sum_{i,j}\left(d_{ij} - x_i - x_j\right)^2}{\sum_{i,j} d_{ij}^2}\right)^{1/2}, \tag{2}$$

where d_{ij} is the ED between the GAP vectors of time series X_i and X_j. In this way, the original raw time series space is largely reduced to two-dimensional space. Each time series X_i is represented as a single data point x_i.

The visualization of MDS shows the distribution of the data in the raw data space to some extent. Here, we used the same technique to show how the adversarial attack works from the data distribution angle. The Coffee dataset is used as an example, and the ResNet is applied as a base neural network. Details are shown in Figure 4.

As shown in Figure 4(a), one can easily separate the set of time series belonging to the two classes by utilizing MDS on the latent representation learned by the network. Yet, in Figures 4(b)–4(d), with the attack ratio eps becoming larger, it becomes harder to classify these two datasets by using linear classifier in the two-dimensional space. With the help of MDS, we could observe that the adversarial attack surprisingly changes the distribution of data.

3.3. Transferability of Adversarial Examples. Transferability is the usual property for adversarial attack examples. The adversarial attack against a neural network can trick neural

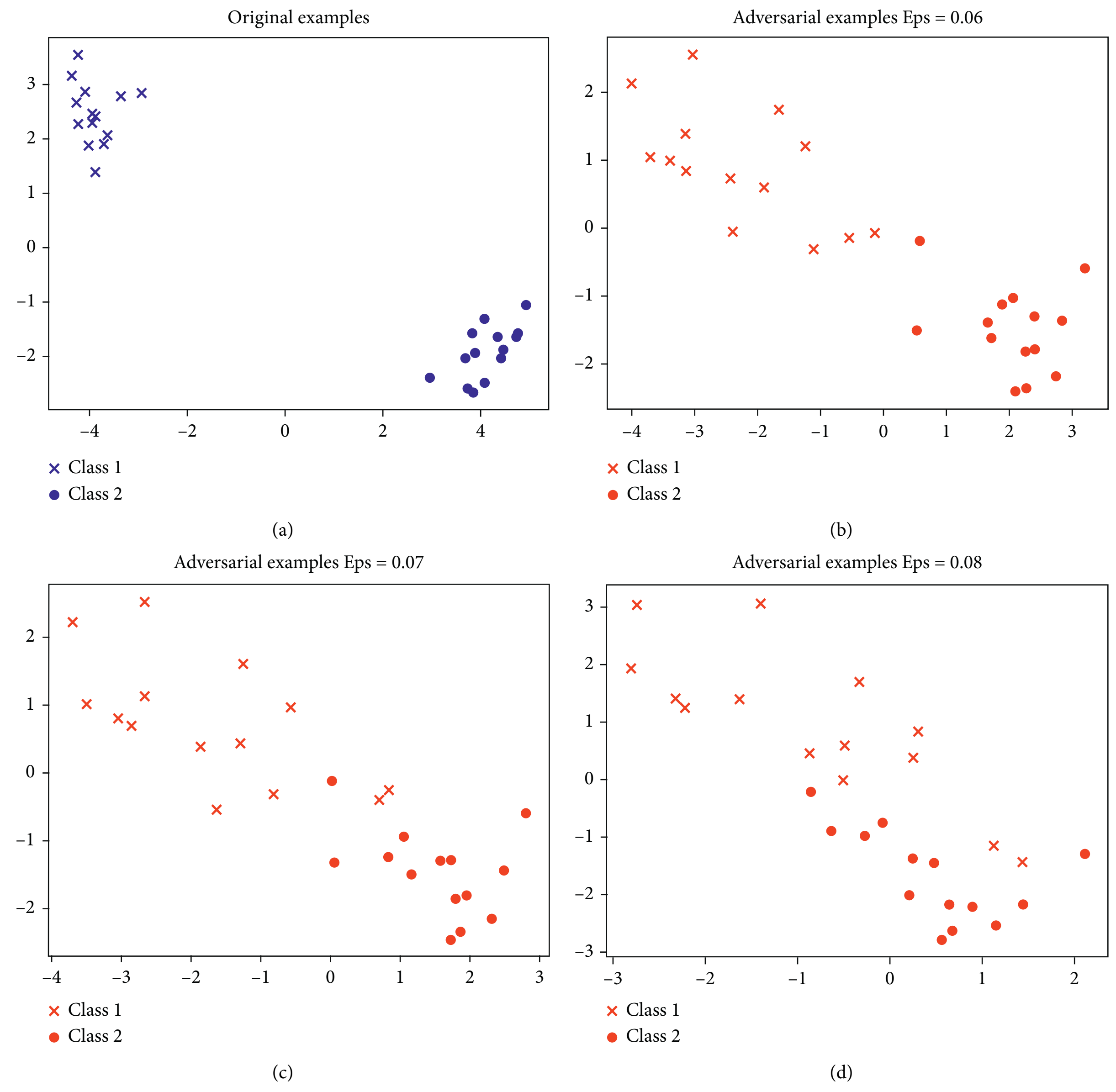


Figure 4: The data distribution of original examples and adversarial examples. (a) The raw time series in ResNet; (b) The adversarial examples in ResNet for eps = 0.06; (c) The adversarial examples in ResNet for eps = 0.07; (d) The adversarial examples in ResNet for eps = 0.08.

networks trained by diverse datasets [23]. Moreover, adversarial attacks for a special architecture can trick other classifiers trained by different machine learning algorithms or even other's neural networks with dissimilar architectures [5].

Recently, Tramèr et al. [24] found that on average, the distance to the model's decision boundary is larger than the distance between two models' boundaries in the same direction which confirms the existence of transferability of adversarial attack examples up to some extent. They also prove, by presenting a counter-example, that transferability is not an intrinsic characteristic of deep neural networks.

Table 1 illustrates that the adversarial examples in time series for one model could also attack other models even they own totally different structures. So, the white-box attack could be launched by generated adversarial examples for other well-known deep neural network models.

As presented in Table 1, the adversarial examples p against FCN could fool ResNet to achieve a low accuracy and vise visa. The experiment shows the transferability of adversarial examples exists in the TSC problem which means the black-box attack could be launched even the details of the backed algorithm are unknown.

3.4. Random Noise Attack. Nguyen et al. [25] uncovered a new type of attack called false positive attack, where adversarial attack examples are misclassified by deep neural networks with the confidence of 99%.

TABLE 1: Accuracy of adversarial examples in different models.

Eps	Clean	Accuracy for ResNet on ResNet adversarial examples	Accuracy for ResNet on FCN adversarial examples	Accuracy for FCN on FCN adversarial examples	Accuracy for FCN on ResNet adversarial examples
0.08	1	0.6429	0.6429	0.4643	0.7142
0.1	1	0.4286	0.9285	0.3929	0.6071
0.12	1	0.3929	0.8571	0.25	0.5357
0.15	1	0.25	0.5357	0.1429	0.4286

Typically, we trained a machine learning model by the following process, as shown in Figure 5; the trained model is deployed to the industry environment after being evaluated on the prepared test dataset. This is extremely dangerous in the environment of IoT due to its device controlling characteristics.

We trained a ResNet model to classify some randomly generated noise data along with the time series data. Unmistakably, the random time series data will be rejected by the classifier with low confidence. However, the random noise data classified were classified as class two with high confidence that prove that there is a potential risk in the model. Some predicted labels of the samples of noise time series examples are visualized in Figure 6.

As illustrated by Figure 6, we notice that even zero values or random noise also can lead to high confidence output. As a result, the model cannot be used directly for intelligent devices.

3.5. Class Activation Map and Adversarial Examples. Class activation map (CAM) proposed by Zhou et al. [26] was exploring to find the discriminative and susceptible field of an image. Later, Wang et al. [9] proposed a one-dimensional CAM application in TSC. Here, we use the CAM method to highlight the susceptible region of a time-series data. Consequently, the susceptible fields of the time-series data are continually distributed which potentially enable that some preprocessing method could improve the robustness of the model.

This method describes the classification of a definite deep learning model to underline the subsequences that contribute the most to a specific classifier. It is to be noted that utilizing CAM is only feasible for the models with a GAP layer prior to the softmax classifier. That is the reason, in this section, we only measured the ResNet model that achieves the highest accuracy for majorities of the datasets. ResNet benefits from the CAM approach using a global average pooling (GAP) layer that helps identify possible regions of an input time series data that contribute to the certain classifier.

Let $A(t)$ be the result of the last convolutional layer MTS with M variables. $A_m(t)$ is the univariate time series for the variable $m \in [1, M]$, where $m \in [1, M]$ is the result of applying the mth filter. Let w_m^c be the weight between the output neuron of class c and the mth filter. As a GAP layer is utilized, therefore, the input to the neuron of class c, i.e., (Z_c) can be computed using the following equation:

$$Z_c = \sum_m w_m^c \sum_t A_m(t). \tag{3}$$

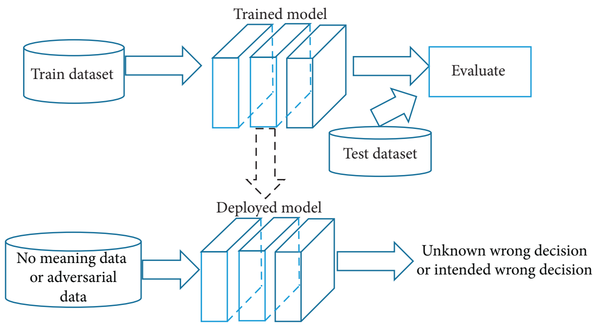

Trained model
Train dataset
Evaluate
Test dataset
Deployed model
No meaning data or adversarial data
Unknown wrong decision or intended wrong decision

FIGURE 5: The process of training a model and the potential threats in the IoT device.

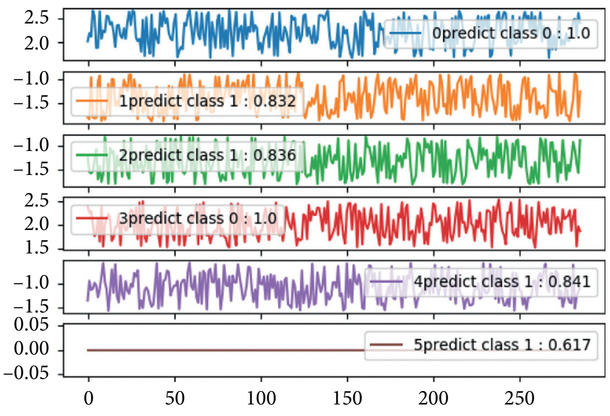


FIGURE 6: The noise data prediction is a valid class with high confidence.

The second summation contributes the averaged time series to the whole time dimension. For simplicity, the denominator is omitted here. The input Z_c can also be represented in equation form as follows:

$$Z_c = \sum_m \sum_t w_m^c A_m(t). \tag{4}$$

Lastly CAM_c, the class activation map, that explains the classification as label c is given by the equation as follows:

$$\text{CAM}_c(t) = \sum_m w_m^c A_m(t). \tag{5}$$

Here, CAM is a univariate time series in which each item at a certain timestamp $t \in [1, \mathrm{T}]$ is equal to the weights being learned by the neural network, i.e., weighted sum of the data points M at time t. Figure 7 shows the result of applying CAM, respectively, on the Coffee dataset.

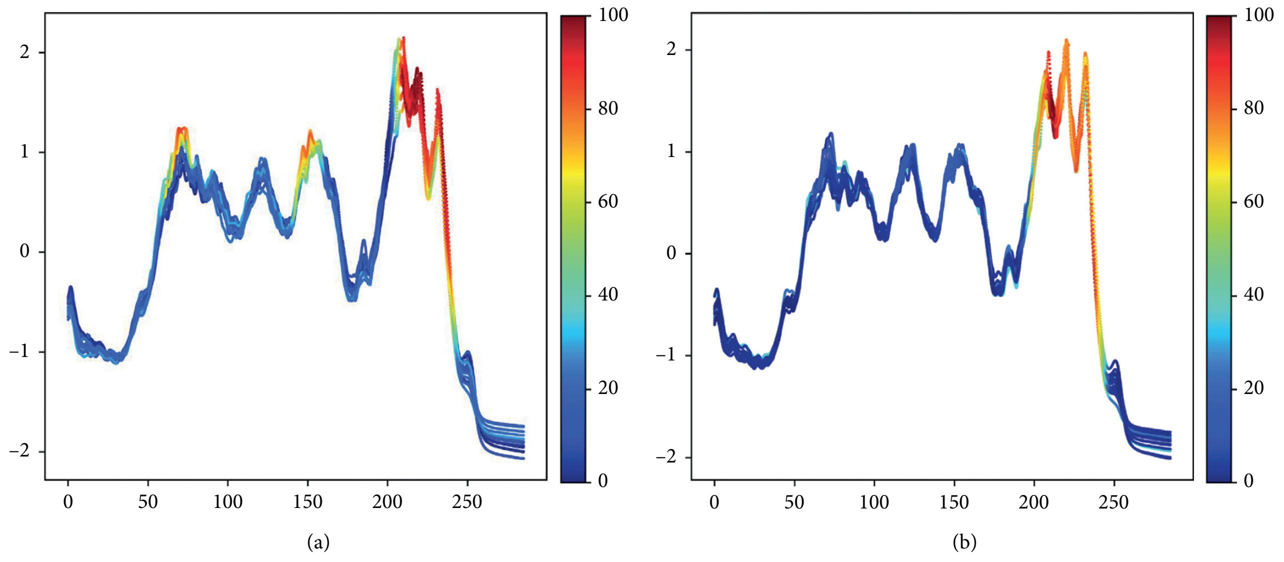


Figure 7: Classification activation map in Coffee dataset. (a) Coffee dataset for class 0. (b) Coffee dataset for class 1.

From Figure 7, we found that the key classification activation fields are in the same points where the vision difference exists. However, the adversarial examples are not trying to modify these places to defraud the classifier.

In Figure 8, we found the adversarial example is a tiny difference from the original time series and the changing place is not in the key area learned by the neural network.

4. Proposed Method

The results of the analysis in Section 3 provides evidence for some potential risks that exist in deep learning models besides the fact that best performance can be achieved in time series classification. Furthermore, in the IoT field, it is extremely dangerous if these algorithms are deployed in devices. We designed a new training strategy based on the encode-decode model to increase the robustness of the model.

Our method consists of two main parts: one is to encode-decode model and the second is a traditional deep neural network model for classification. In the encode-decode model, the input is the nonlinear transformation of the original time series. Here, we applied the thermometer encoding method as the nonlinear transformation. The decoded output is the original time series that is recovered from its thermometer encoding forms. The reason to use the encode-decode model is to take advantage of its nonlinear transforms to remove some noise and adversarial perturb which is based on linear gradient signs. The schema of our proposed method is shown in Figure 9.

Our method tries to bring nonlinear transformation by the encode-decode model which will defend the traditional adversarial attacks. The network consists of two main parts, one is encode-decode part. In this part, the network tries to learn a noise and nonlinear function which tries to minimize the loss of original example with the thermometer discretized examples.

The encoder maps the input to a fixed-length vector (which needs to contain all the input information) and the decoder then outputs the translation. In the model, the encoder learns a coding sequence representing the semantic information of time series, and the decoder maps the sequence to the original time series.

First, the time series will be discretized into an average of ten evenly spaced levels. Additionally, the thermometer encoding method is applied to the discretized curves. Based on the thermometer encoding time series, the encode-decode model is trained to reconstruct time series. Therefore, the loss function we used here is the mean-square error (MSE).

In the process of training the encode-decode model, we add some random noise to the time series that increase the reconstruction ability.

Figure 10 shows that the encoder part tries to learn some robust illustration of the input time series. The decoder tries to map the input to its original time series. Here, we add some random noise to the original input time series to increase the robustness of the encode-decode model.

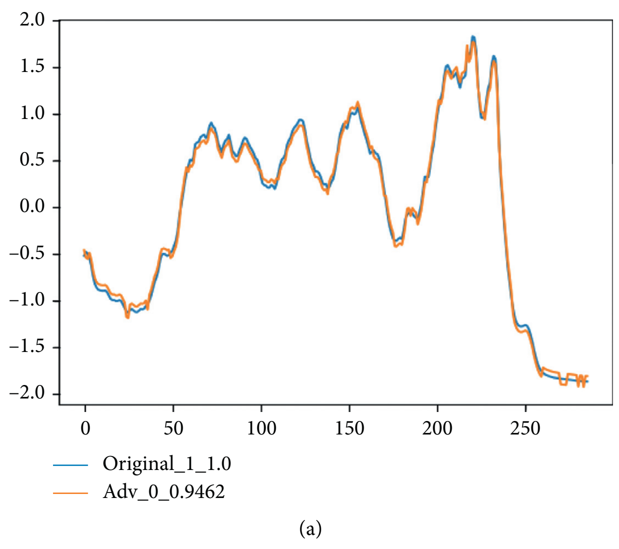
In order to discretize the input time series x without losing the relative distance information, Buckman [3] proposed thermometer encodings. For an index $j \in \{1, \ldots, k\}$, let $\tau(j) \in \mathscr{R}^k$ be the thermometer vector defined as follows:

$$\tau(j)_l = \begin{cases} 1, & \text{if } l \geq j, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Then, the discretization function f is defined for a time index point $i \in \{1, \ldots, n\}$ as follows:

$$f_{\text{therm}}(x)_i = \tau(b(x_i)) = \text{Sum}(f_{\text{onehot}}(x_i)), \tag{7}$$

where Sum is the cumulative sum function and $f_{\text{onehot}}(x_i)$ is the one-hot coding method. The thermometer encoding



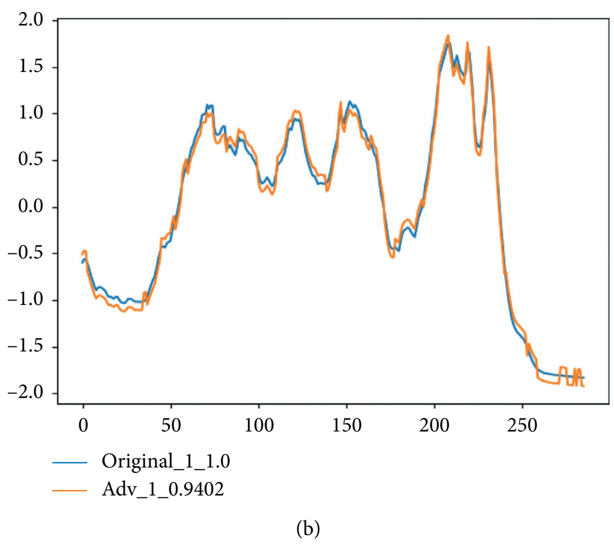


FIGURE 8: Two typical adversarial examples in the Coffee dataset.

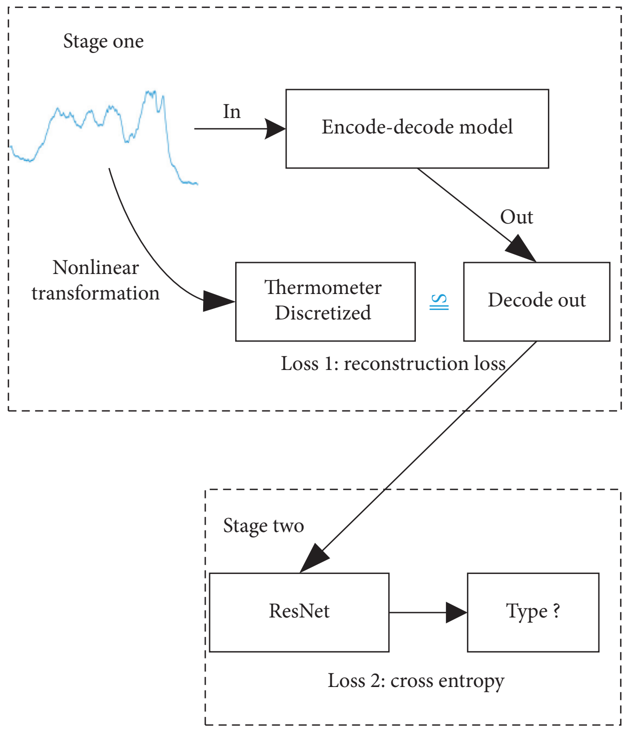


FIGURE 9: The procedure of the proposed method.

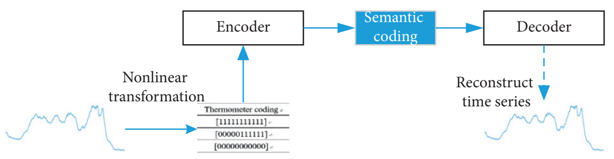


FIGURE 10: The encode-decode model in the proposed method.

could retain the order information for the time series, i.e., for pixels i and j, if $b(x_i) \neq b(x_j)$ and $x_i < x_j$, then $\tau(b(x_i))_2 < \tau(b(x_j))_2$.

This characteristic is very important for time series that hold the order and shape information of the original time series. Figure 11 shows the discretize process of a time series. Table 2 shows the thermometer encoding result of a continuous value.

The time series can be disseized by the average bin method and transformed into other code. The coding method is highly nonlinear, which could defense the attack for the gradient-based attack method.

The curve with certain noise can be restored normally after discretization and encode-decode model. The well-trained encode-decode model could recover the original time series from its thermometer encoding. We showed the example of the Coffee dataset to illustrate its effectiveness in Figure 12.

As illustrated in Figure 12, the reconstruction time series contains all the information of the original time series. The difference is the high-frequency part of the time series, which looks to link random noises. We showed that the deep learning model trained on these reconstruction examples could show high accuracy and ability to defend from adversarial attacks.

5. Experiment and Evaluation

In this section, we present an attack method FGSMs and ResNet [21] architecture. We then use FGSMs to generate adversarial time series attack examples for the ResNet model.

5.1. Data Sets and Comparison Method.

85 datasets of the UCR archive are utilized in experiments [27]. These datasets encapsulate diverse time series data from fields like electricity industry, food security, image, and sensors.

One of the dataset is electronic devices known as smart meters, which record detailed electricity consumption data. A previous study [28] showed that these electricity data could be used to analyze the type of electric device. The

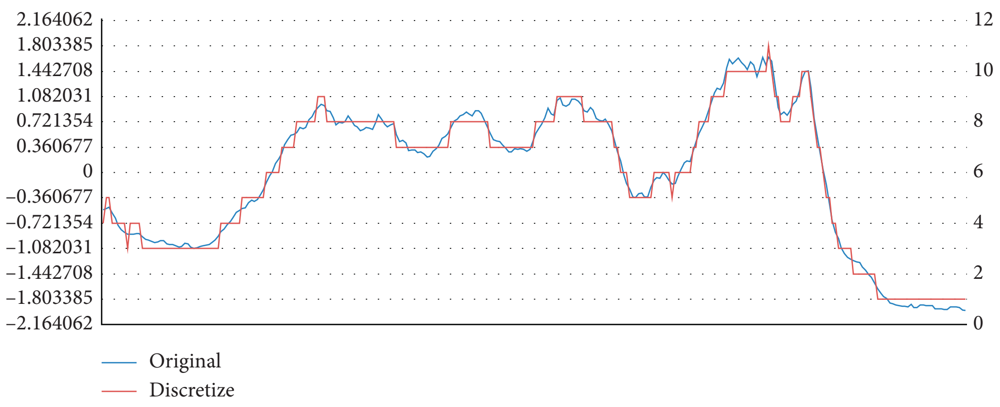


Figure 11: Examples of mapping continuous-valued inputs to quantized inputs and thermometer codes with ten evenly spaced levels.

Table 2: Examples of mapping continuous-valued inputs and thermometer codes with ten evenly spaced levels.

Continuous value	Quantized	Thermometer encoding
0.13	0	[11111111111]
0.54	0.5	[00000111111]
0.96	1	[00000000000]

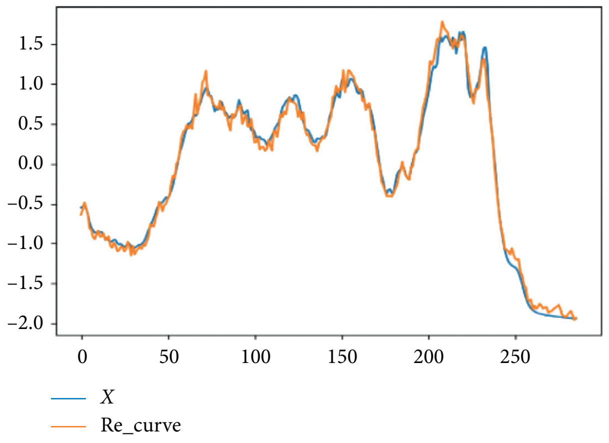


Figure 12: The reconstruction curve from the original time series.

consumption time series could be modified by adding some small perturbation to misguide the classifier in the device. The aim to collect and analyze the electricity consumption data is to monitor the device being used by the citizens' homes and in future to reduce carbon footprint. 375 univariate time series come under the umbrella of the dataset. The classes are Microwave, Toaster, and Kettle of length 720. The dataset classically illustrates IoT time series attack example and is a vital task in the intelligent device.

ResNet architecture the same as [8] has been employed for the comparison process. Details about the architecture and its parameter are shown in Table 3.

The block of ResNet is illustrated in Figure 13.

In ResNet, time series act as input and the possible classes K serve as an output. The convolution kernel size is 8, 5, and 3 for every individual block of the ResNet which indicates that for extracting some useful features, it will have the neighbor size 8, 5, and 3. The ResNet we employed, comprises of three blocks, and they have 64, 128, and 128 filters, respectively.

Table 3: Layers details in one block of the ResNet.

name	Layer	Parameter
Conv_x	Conv1D	Filters = 64, kernel size = 8, stride = 1
Conv_x	BatchNormalization	
Conv_x	Activation	Function = ReLU
Conv_y	Conv1D	Filters = 64, kernel size = 5, stride = 1
Conv_y	BatchNormalization	
Conv_y	Activation	Activation function = ReLU
Conv_z	Conv1D	Filters = 64, kernel size = 3, stride = 1
Conv_z	BatchNormalization	

5.2. Result and Analysis of Attack and Defense.

The experiments are conducted on Keras 2.1 and TensorFlow 1.8. The number of samples in training and testing phase is decided by the original public available dataset (UCR). We trained the encode-decode ResNet network and extracted the ResNet part as the attack target. The input of the attack model is the original time series; the gradient of this model is computed in the same way as illustrated in the work. Although in our method, we used a thermometer as the input to train the encode-decode model, and the comparison model is the same ResNet as illustrated in [8]. The experiments in this manuscript are carried out to show the efficiency of the encode-decode model in the defense part. The results of the defense are shown in Table 4. During the attack and defense stage, the perturbation ratio ε is set to 0.1.

In Table 4, we could see that the accuracy of most of the datasets is largely improved compared with encode-decode training. The result shown in Table 4 reveals that our method could resist the attack of FGSM in the TSC problem to some extent.

To future analyze, the encode-decode model could defend against the attack by FGSM, and we get the accuracy of a typical sensor dataset, i.e., Coffee dataset under different ε. First, we generate some adversarial examples using FGSM, and then the time series examples are smoothed or

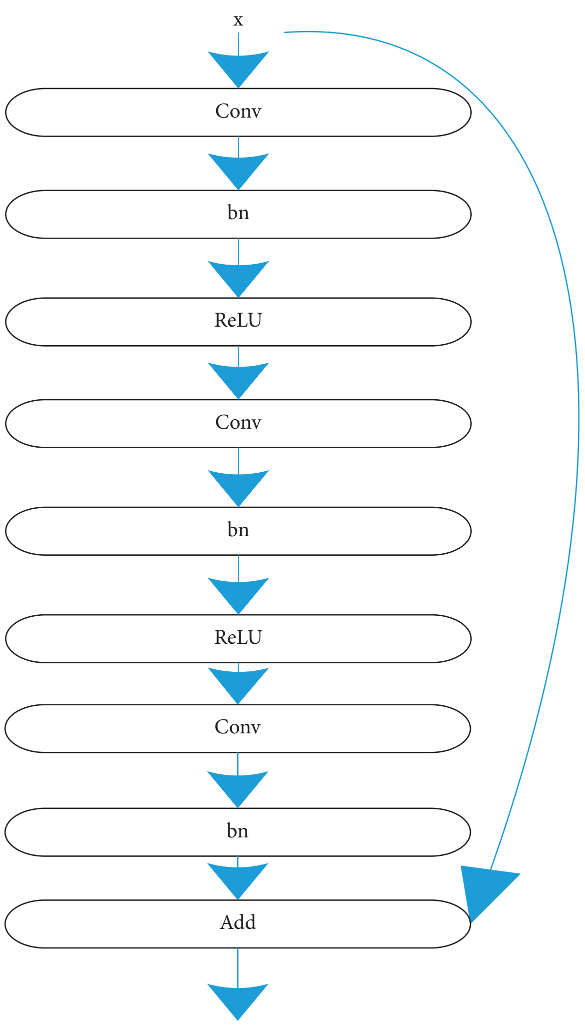


Figure 13: The block building in ResNet.

Table 4: Defense results of our method for ResNet and FGSM.

Dataset	ResNet_ori (Fawaz, et al, 2019)	ResNet_fgsm_adv (Fawaz, et al. 2019)	Encode-decode ResNet_fgsm_adv
50words	73.2	17.1	**49.22**
ADIAC	83.1	3.1	**10.16**
ArrowHead	85.1	33.1	**75**
Beef	76.7	20	**30**
BeetleFly	85	15	**60**
BirdChicken	95	55	**65**
Car	93.3	21.7	**35**
CBF	98.9	86.1	**98**
Coffee	100	50	**75**
SmallKitchenAppliance	78.9	40.5	**65.4**

transformed in different ways. Second, these processed examples are evaluated on these models. Details of accuracy on the Coffee dataset are shown in Figure 8. Coffee dataset [20] is a two-class problem that discriminates between Arabica and Robusta coffee beans.

The encode-decode ResNet shows a good defense result for this dataset, and the accuracy curve is shown in Figure 14. In this figure, we could find that the accuracy of these two datasets decreased slowly as the number of perturbation increases. It means the attack of FGSM still works here, but its effectiveness is largely reduced. The reason lies in the thermometer encoding because it is a highly nonlinear transformation. The thermometer encoding discretizes the time series and retains the order information about the original curve.

5.3. Preprocessing Method for Defense Adversarial Attack. Actually, in an industrial environment, we could apply some practical preprocessing methods such as time series smooth method to weaken the fluency of adversarial examples. Here,

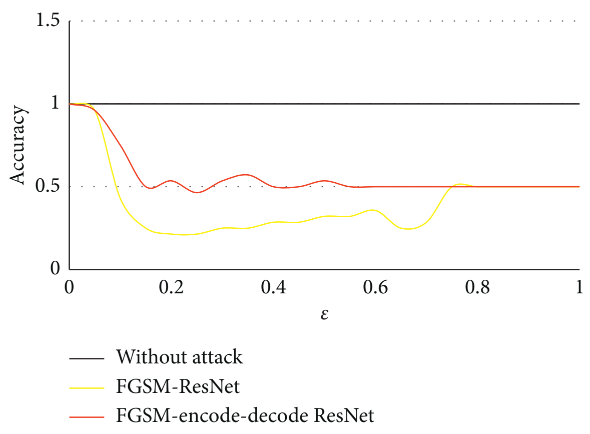


Figure 14: The accuracy of Coffee dataset in encode-decode ResNet with FGSM attack in different perturbation ratios.

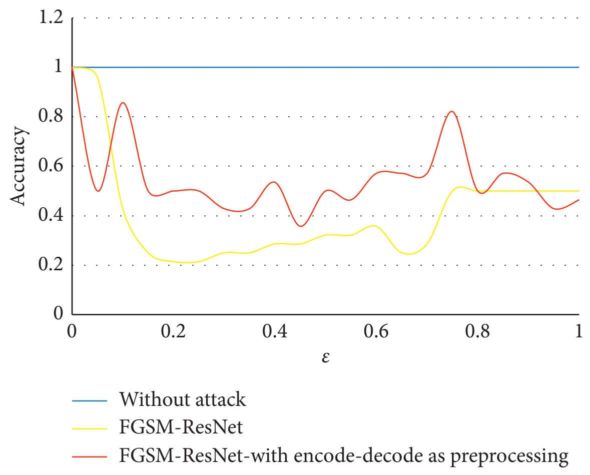


Figure 15: The accuracy of Coffee dataset in ResNet with the encode-decode model as a preprocessing method for defense FGSM attack.

we show two methods known as smooth and encode-decode to assist the attack.

In the experiment, we first applied the thermometer encoding method to transform the adversarial examples; then, the encode-decode model is used to map the thermometer encoding back into the original time series. Of course, the reconstructed time series is different from the original time series. The recognition accuracy is shown in Figure 15.

As illustrated in Figure 15, the yellow line is below the red line, which means the encode-decode model improves the accuracy of attacks by the FGSM. This result hints that the encode-decode model could be used as a data preprocessing method before being put into the classification model.

6. Conclusions

The proposed method of this paper is of using encode-decode model joint training strategy to strengthen the robustness of the deep learning model. The experiments reveal that our model can resist FGSM attacks to some extent. Moreover, the encode-decode model could be used as a way of preprocessing to weaken the attack from FGSM.

Though, it is not easy to eliminate the white-box attack launched by FGSM. Our method improves the robustness of the trained model but fails to resist the attack completely. To check the effectiveness of our method, more experiments on other datasets are required as well.

Moreover, we found that different trained models own different power against the same attack, and it is hard to evaluate the goodness of the model. Fundamentally, there are no theoretical studies on how to quantify the goodness or robustness of a trained model. Therefore, given the popularity of applying the deep learning method to IoT data analysis, it still needs more research to focus on the interpretability of deep learning models. Our future research directions include how to evaluate the defensive capability to adversarial examples in the area of IoT data.

Data Availability

The data used to support the findings of this study are included in the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Projects of International Cooperation and Exchanges NSFC (grant no. 62061136006), Beijing Natural Science Foundation (no. 4202021), and Scientific Research Initiation Funds (grant nos. 2462020YJRC001 and 110051360002).

References

[1] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, “Internet of things security and forensics: challenges and opportunities,” *Future Generation Computer Systems*, vol. 78, 2018.

[2] S. M. Abdelfattah, G. M. Abdelrahman, and M. Wang, “Augmenting the size of EEG datasets using generative adversarial networks,” in *Proceedings of the International Joint Conference on Neural Network*, pp. 1–6, Rio de Janeiro, Brazil, August 2018.

[3] M. Shafiq, Y. Z. Tian, and M. X. GuizaniDu, “Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city,” *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[4] M. Shafiq, Z. Tian, A. K. Bashir et al., “CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine learning techniques,” *IEEE Internet of Things Journal*, vol. 14, no. 99, pp. 1606–1615, 2018.

[5] N. Papernot, P. D. Mcdaniel, and I. J. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” *Cryptography and Security*, vol. 2016, 2016.

[6] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, “Deep learning for time series classification: a

review," *Data Mining and Knowledge Discovery*, pp. 1–47, Springer, Berlin, Germany, 2019.

[7] J. L. Agnieszka Nawrocka, "Determination of food quality by using spectroscopic methods," in *Advances in Agrophysical Research*, S. G. A. A. Stepniewski, Ed., IntechOpen, London, UK, 2013.

[8] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, "Adversarial attacks on deep neural networks for time series classification," 2019.

[9] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: a strong baseline," in *Proceedings of the International Joint Conference on Neural Network*, pp. 1578–1585, Anchorage, AK, USA, May 2017.

[10] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: attacks and defenses for deep learning," *IEEE Transactions on Neural Networks*, vol. 30, no. 9, pp. 1–20, 2019.

[11] K. Eykholt, I. Evtimov, E. Fernandes et al., "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, Salt Lake City, UT, USA, June 2018.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[13] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proceeding of the International conference on learning representations*, Toulon, France, April 2017.

[14] N. Papernot, P. D. Mcdaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 582–597, San Jose, CA, USA, May 2016.

[15] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of JPG compression on adversarial images," in *Proceedings of the Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.

[16] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," in *Proceedings of the International Conference on Learning Representations*, Banff, Canada, April 2014.

[17] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, October 2017.

[18] A. S. Ross and F. Doshivelez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 1660–1669, New Orleans, LA, USA, October 2018.

[19] J. Buckman, A. Roy, C. Raffel, and I. J. Goodfellow, "Thermometer encoding: one hot way to resist adversarial examples," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, May 2018.

[20] R. Briandet, E. K. Kemsley, and R. H. Wilson, "Discrimination of Arabica and Robustain instant coffee by fourier transform infrared spectroscopy and chemometrics," *Journal of Agricultural and Food Chemistry*, vol. 44, no. 1, pp. 170–174, 1996.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[22] J. B. Kruskal and M. Wish, "Multidimensional scaling," 1978.

[23] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," in *Proceedings of the International Conference on Learning Representations*, Banff, Canada, April 2014.

[24] F. Tramer, N. Papernot, I. J. Goodfellow, D. Boneh, and P. D. Mcdaniel, "The space of transferable adversarial examples," *Machine Learning*, vol. 2017, 2017.

[25] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: high confidence predictions for unrecognizable images," in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 427–436, Boston, MA, USA, June 2015.

[26] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 2921–2929, Las Vegas, NV, USA, June 2016.

[27] A. Bagnall, H. A. Dau, J. Lines et al., "The UEA multivariate time series classification archive, 2018," 2018.

[28] P. O. A. R. Foreman, *Powering the Nation: Household Electricity Using Habits Revealed*, Energy Saving Trust/DECC/DEFRA, London, UK, 2012.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6677027, 10 pages
https://doi.org/10.1155/2021/6677027

Research Article

An Anomaly Detection Algorithm Selection Service for IoT Stream Data Based on Tsfresh Tool and Genetic Algorithm

Zhongguo Yang,[1] **Irshad Ahmed Abbasi**,[2] **Elfatih Elmubarak Mustafa**,[2] **Sikandar Ali**,[3,4] **and Mingzhu Zhang**[1]

[1]*School of Information Science and Technology, Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, North China University of Technology Beijing, Beijing, China*
[2]*Department of Computer Science, Faculty of Science and Arts at Belgarn, P.O. Box 60, Sabt Al-Alaya 61985, University of Bisha, Saudi Arabia*
[3]*Department of Computer Science and Technology, China University of Petroleum-Beijing, Beijing 102249, China*
[4]*Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum-Beijing, Beijing 102249, China*

Correspondence should be addressed to Sikandar Ali; sikandar@cup.edu.cn

Received 31 December 2020; Revised 21 January 2021; Accepted 23 January 2021; Published 8 February 2021

Academic Editor: Shah Nazir

Anomaly detection algorithms (ADA) have been widely used as services in many maintenance monitoring platforms. However, there are numerous algorithms that could be applied to these fast changing stream data. Furthermore, in IoT stream data due to its dynamic nature, the phenomena of conception drift happened. Therefore, it is a challenging task to choose a suitable anomaly detection service (ADS) in real time. For accurate online anomalous data detection, this paper developed a service selection method to select and configure ADS at run-time. Initially, a time-series feature extractor (Tsfresh) and a genetic algorithm-based feature selection method are applied to swiftly extract dominant features which act as representation for the stream data patterns. Additionally, stream data and various efficient algorithms are collected as our historical data. A fast classification model based on XGBoost is trained to record stream data features to detect appropriate ADS dynamically at run-time. These methods help to choose suitable service and their respective configuration based on the patterns of stream data. The features used to describe and reflect time-series data's intrinsic characteristics are the main success factor in our framework. Consequently, experiments are conducted to evaluate the effectiveness of features closed by genetic algorithm. Experimentations on both artificial and real datasets demonstrate that the accuracy of our proposed method outperforms various advanced approaches and can choose appropriate service in different scenarios efficiently.

1. Introduction

With the growth of the Internet of Things (IoT), the sensor or stream data is bound to be collected at tremendous speed. In such real-time scenarios, there can be various anomalous data streams, for example, the data diverge from the usual behavior of the stream or the abruptly jumped data [1], which are dissimilar to familiar patterns.

It is critical for further decision making to capture these anomalous data accurately and timely. Banerjee et al. [2] introduced the trend of everything as a service (XaaS). Following Banerjee et al. [2], a lot of researchers try to encapsulate various data or common functions into services. For example, streaming as a service is studied by many researchers [3–5], which can provide the sharing and simple processing capabilities for stream data. The idea of choosing suitable service or methods can be referred to in [6–8]. It is proposed to provide common functions for various data sources, which enable users to conveniently reuse these functions and form more complex functions through service composition.

In real-world software systems, numerous anomaly detection algorithms (ADAs) are industrialised and are offered as a service to be utilised in diverse domains [9, 10]. In our preceding work [11], a proactive data services abstraction was applied to appropriately encapsulate present ADAs into a service.

Even though, with the scenario in hand, it is still a challenge to effectively capture anomalous data considering various circumstances. Following the concept of the No-Free-Lunch (NFL) optimisation theorem [12], it is infeasible to find a single algorithm for all the cases that dominate all others on the same optimisation problem [1]. In the state-of-the-art survey paper, Braei and Wagner [13] state that for the most part the univariate dataset may suffer from contextual anomalies; therefore, statistical methods will not perform well. Deep learning models may perhaps increase the area under the curve (AUC) and neural network models might outperform the statistical methods. On contrary, the volume of novel stream data can appear frequently and continuously and can result in missing part of the anomalous data through manual service selection. Consequently, running an ADS possibly will not adjust to different types of stream data. Therefore, for faster and more accurate anomaly detection, it is obligatory to choose an appropriate service for different stream data dynamically at run-time.

Since each type of anomaly detection algorithms gives better results only for a particular set of stream data [14]. Therefore, to automatically choose appropriate services for diverse IoT scenarios, it is required to correctly and quickly characterise the underlying stream data. Hence, proper service might be chosen and configured based on the pattern of a particular stream of data. Keeping in view the gigantic volume of stream data, this study finds out that several IoT streams are alike owing to their shape similarities and implicit relations.

For effective handling of anomalies from various stream data, based on the above observation, in this paper, an Anomaly Detection via Service Selection (ADSS) framework was proposed. To recognise the pattern of various stream data, in our proposed ADSS framework, it tries to capture intrinsic similarity and dissimilarity in various stream data established on time-series statistical features. Moreover, a fast classifier based on the XGBoost algorithm is trained to record features of stream data in order to detect appropriate ADS dynamically at run-time. Due to the presence of the best classifier, our ADSS method can identify the dynamics of data stream patterns of newly appearing stream of data and then choose and configure the suitable service.

Firstly, it is well known that there could not be an algorithm that could defeat others in all the datasets. Consequently, our aim of this study is not to build a model or to develop a new algorithm which could beat all the other algorithms in all the datasets. Instead, a method is designed to capture the variation of the stream data in the run-time and configure different algorithm to handle the stream data. Experimental results show that we could achieve a better performance in the long run.

This study focuses on the selecting algorithms based for dynamically changing IoT stream data. The original idea is to construct features to be a representation of different stream data and build a supervised model to recommend a suitable algorithm for a certain stream data. Collections of historical data are gathered from a real monitor system. Further, on the basis of these data, an XGBoost model is trained based on the feature and its label. Here, the label is the best algorithm which is more suitable for a certain kind of stream data. This manuscript is the extended version of our recently published conference paper [15].

In the revised version of the manuscript, the features' construction process is improved by applying a Tsfresh tool and intelligent optimisation algorithm [16]. The former tool is taken to extract multiple features of time-series. These features consist of 100+ kinds of features from different angles which could represent intrinsic features of stream data completely. Moreover, an intelligent optimisation algorithm such as genetic algorithm is applied to help choose a subset of features which could further result in the reduction of computing complexity of the algorithm recommendation procedure. The specific contributions of the manuscript are summarised below:

(i) In this paper, we develop a method that facilitates IoT-based systems to automatically choose appropriate services using the existing data features in order to detect an anomaly.

(ii) In this paper, we develop a service update framework in which service quality and its resultant algorithms and data stream are recorded. The aforesaid historic data will assist in the training of different decision models that paves the approach for accurately recommending ADS. In this approach, freshly designed algorithms can easily be added to the service pool.

(iii) In this paper, we carried out various experiments by means of data streams from NAB [17] and Yahoo datasets [18]. The experimental results demonstrate that our method can select the best service dynamically according to changes in the stream data pattern.

(iv) In this paper, an improved features' construction method by applying Tsfresh tool and intelligent optimisation algorithm is devised [16].

The remainder of this article is organised as follows. Section 2 describes the related work to build a proper problem statement. The proposed ADSS framework is accessible in Section 3, while Section 4 is based on experimental outcomes. Section 5 is the last section which summarises the paper.

2. Background and Related Work

2.1. Anomaly Detection Algorithms. In this study, the unsupervised methods are mainly considered to detect anomalies due to its good generalization ability. The possible reasons why we do not consider supervised methods are as follows. Firstly, in real-time IoT arrangements, different types of time-series data are collected that are hard to label for anomalies. Secondly, to rapidly deploy ADS, almost there is very less or no time to train a complex anomaly detection model. Thirdly, for the dynamic change of time-series in real-time IoT systems, even some of the good models perform badly and cannot handle this dynamism. Summary of the unsupervised class of ADAs is given in Table 1.

Table 1: Summary of stream data anomaly detection algorithms.

Typical algorithms	Category	Characteristic and limitations
Prediction confidence interval (PCI) for time-series outlier detection, simple exponential smoothing (SES) [19], and ARIMA model [20]	Statistical approaches	(1) A supposition about outlier data and normal data need to made first (2) Domain-specific knowledge is needed for threshold selection depends on
Autoencoder [21], LSTM [22]	Artificial neural computing	Since clustering methods cannot deal with continuous changes in data, therefore careful parameter tuning is needed
Density-based spatial clustering of applications with noise (DBSCAN) [23], subsequence time-series clustering (STSC) [13], isolation forest [24], local outlier factor (LOF) [25], one-class support vector machine (OC-SVM) [26]	Machine learning approaches	Work on stream data; therefore, the normal reference model might be outdated at the moment they are actually used

Although, for anomaly detection, there are numerous deep learning algorithm-based methods, for example, AutoEncoder [21] and LSTM [22], they cannot be used directly on the continuous stream data, because these methods need fine parameter tuning and a lot of training data. Allowing for the scenario of frequent changes of data pattern or anticipated behavior in the frequently launched streams stream, the notion of selecting appropriate service algorithms is becoming challenging.

2.2. Anomaly Detection Algorithm in the System. To present a unified and easier way to adapt to changes and accurately detect anomalies in diverse circumstances, a lot of ADAs are delivered as a service.

In [14], the first ever ADS framework was developed to consider the aforementioned problems through semi-supervised learning and clustering. This study was the first work that applies semisupervised learning to key performance indicator (KPI) anomaly detection [14]. Still, the postulation of huge resemblance in KPI stream data is not effective in conventional IoT stream data.

In [10], an anomalous behavior recognition system composed of two phases was developed based on the past data learning the normal behavior of the system in the first phase and then by processing real-time data and detecting abnormal behavior in the system dynamically in real time in the second phase. In their system, complex event processing (CEP) patterns and anomaly detection are combined as a REST service to be utilised through the interface by a user.

In [27], the authors divided stream data into four different time-series groups, i.e., periodic, stationary, non-periodic, and nonstationary. Furthermore, they used diverse techniques to detect anomalous data.

In [28], the authors state that, in the age of big data, it is a very challenging but important task to detect anomalies. They presented the Interactive Data Exploration As-a-Service approach for the identification of significant data.

A dynamic IoT stream data ADA must recognise various data pattern changes in diverse stream data anomaly detection approach. Though previously researchers were aware of the problem of runtime outlier detection, yet solution formation did not consider this problem and ignored consequent changes in the stream data. While working with a fast growing volume of IoT data with their respective dynamic nature, current approaches are not effective.

We attempt to develop a framework based on the features collected in the first phase to characterise the time-series data and then apply deep learning models in the second phase to recognise the pattern of data that will help reconfigure the ADS dynamically in the run-time.

3. Framework for IoT Stream Data ADS Selection

3.1. Description of Our Proposed ADS Framework. The framework developed in this paper comprises of three parts: (i) service selection procedure, (ii) encapsulations of ADAs, and (iii) service applied procedure. Many publicly available unsupervised ADAs are incorporated for the development of ADAs. As mentioned before, the available ADAs can be encapsulated into services based on PD-service abstraction. A RESTful API is used for the selection of ADS. Service receiver can define individual views to build IoT applications and can get the anomalous data via the Uniform Resource Identifier (URI) of service. The entire working of the developed ADSS framework is illustrated in Figure 1. As shown in Figure 1, the collected tuples are portions of historical data that can be collected through recording the stream data for a long time by field experts along with the appropriate ADAs.

Stream data along with its appropriate algorithm are kept in the database in the form of a tuple **<stream data, algorithm>** that can be used as a metadata for onward service identification and selection procedure. These historic data can be updated by collecting running examples from anomaly detection systems or by experts in this field. Usually, these recorded data monitored the performance of various ADAs and stream data that can be used to generate a scheme to select an ADS for specific stream data.

ADSS is the basic unit of our framework. Each stream data can be represented by a feature vector for by applying a stream feature extraction technique on stream data. As a training data of service selection model, the paired data are constructed and combined with the best possible service. In the service applying part of our framework, a new stream data is transformed to features vector grounded on the same

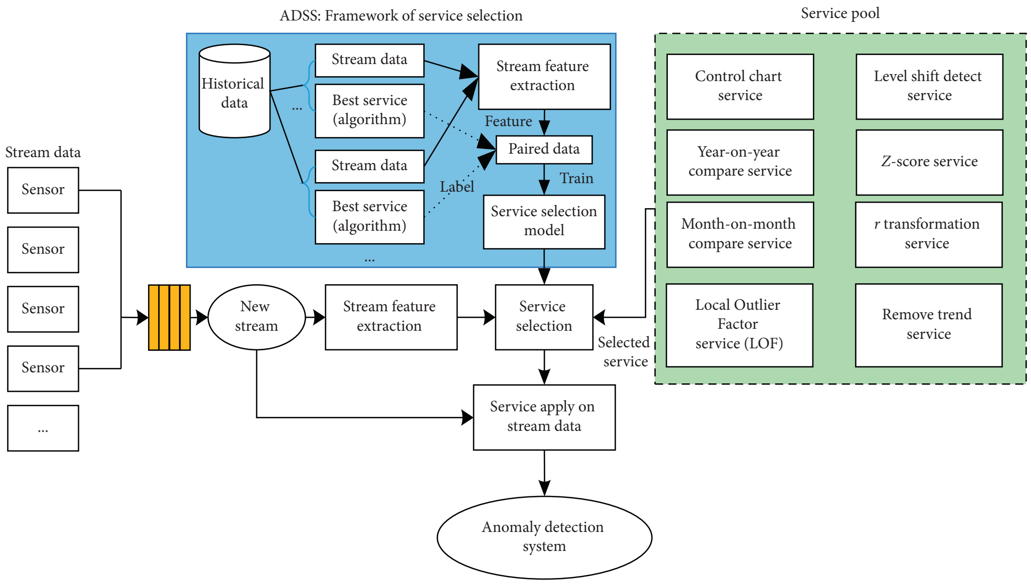


Figure 1: Anomaly detection via service selection framework for service selection.

feature extraction technique. Finally, the service selection model is used on the feature vector to select appropriate service for the existing stream data and ultimately call the service in real time to identify anomalous data.

3.2. Model for Service Selection. Abundant stream data are gathered and their feature is extracted through the process discussed in the previous sections in order to select appropriate services for stream data anomaly detection. The finest ADS is chosen by analysing the historic data based on the recorded stream data fragment and its corresponding best service. In general, few ordinary services are tested repeatedly on these stream data fragments to identify its finest service. Grounded on the stream data fragments and its finest ADS, the service selection problem has been transformed into a pattern recognition problem.

Taking into account its computing efficiency, in this paper, XGBoost [29] is utilised as a base classifier to choose a service for real-time stream data anomaly detection. This procedure is illustrated in Figure 2. It should be noted that any classifier can be used in our framework. However, in this study, we have chosen the XGBoost algorithm as to best choose service considering the easy explanation and high computing efficiency of the XGBoost algorithm.

The time-series features are the main part of our framework, as presented in Figure 2. Some renowned stream data features are taken from publicly available features and some former anomaly detection schemes. What is more, a feature selection method was employed to find some good features to capture stream data essential features. The objective of the selected features of stream data is to accurately and quickly select the appropriate service dynamically in real time for novel evolving stream data.

3.3. Stream Data Patterns Representations. Stream data may generate dissimilar patterns as demonstrated in Figure 3. According to Bu et al. [14], supervised techniques such as SVM or deep learning-based techniques are not achievable for the huge amount of novel IoT stream data applications and the dynamic nature of the stream data. This might be due to two reasons: difficult parameters tuning process and a large amount of training data.

Researchers like Bu et al. [14] state that for some kinds of stream data simple ADAs may perform well compared to some multifaceted algorithms such as deep learning. The pattern of stream data can also be recognised in time which overlays the way for future algorithm selection in modern microservice architecture also recognised as a service selection. The main contribution of our work focuses on the extraction of features to characterise stream data and based on these features select suitable algorithm service.

In order to select useful features that could distinguish different stream data patterns, a feature selection method was applied. We surveyed all the features which could be considered for the representation of time-series data. There are multiple types of features from different angles such as statistics, mathematics, shape, distribution of data, and others in the classification of time-series field.

Christ et al. [16] automatically extract 100 features from time series and develop a tool called Tsfresh. These features label basic characteristics of the time series, for example, maximal or average value, the number of peaks, and additional complex features, for example, time setback symmetry statistics. At the same time, through hypothesis testing to reduce the characteristics to those, which can best explain the trend called decorrelation. These feature sets are then used to construct machine learning or statistical models based on time series data such as classification or regression tasks.

In addition, these collected features are the reflection of the inherent nature of data patterns, for example, the distribution, the fluctuation, and shape of data. Some typical features are demonstrated in Figure 4.

As is shown in Figure 4, these simple features or complex features are designed to characterise the time-series data

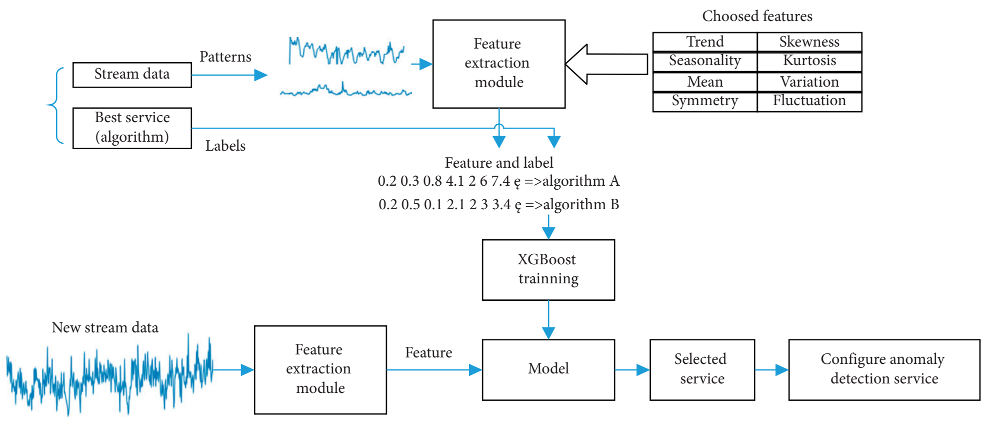


Figure 2: The process of service selection based on feature extraction.

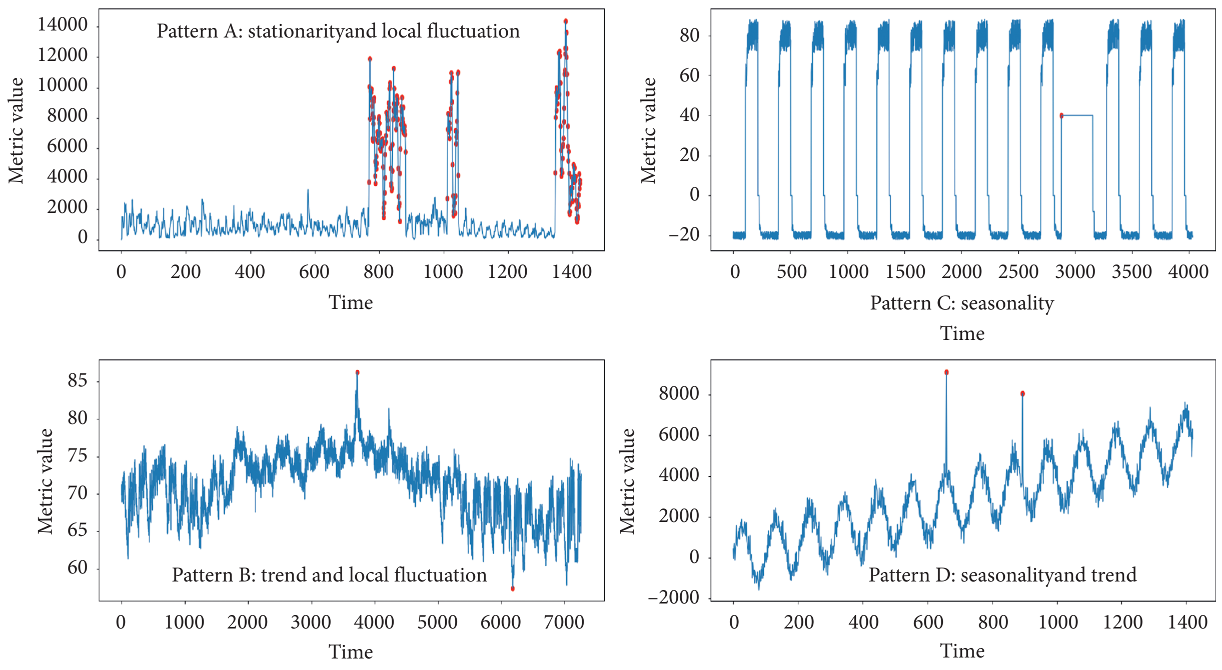


Figure 3: Behavior characteristic of time-series in real-world sensor datasets. Anomalous data characteristics are signposted in red. The time-series data are taken from Yahoo! datasets [18] and NAB [17].

from different angles and own their special geometric interpretation or statistical meaning. These special characteristics are quantified by computing these features. In other words, it is possible to distinguish these stream-data from each other by comparing these features. More details about other features in Tsfresh are discussed in Table 2.

As is presented in Table 2, some computing techniques are taken from Extendible Generic Anomaly Detection System (EGADS) [30], and some metrics are taken from Tsfresh [16] and the rest from other renowned statistical techniques such as standard deviation and mean. Local fluctuation, metrics of symmetrical values, and fluctuation ratio are recommended in our study to characterise stream data from diverse perspectives.

The flowchart of selecting features from multiple original features is illustrated in Figure 5. As shown in Figure 5, the genetic algorithm (GA) [31] is applied to find a feature subset, which is enough to characterise different traits of various stream data.

In the process of GA, the fitness computing consists of two steps: decoding individual to feature subset and computing test score based on the feature subset. The test score is utilised as the fitness of the individual. The other steps of GA such as selection, crossover, and mutation are following the normal behavior as in the traditional computing processes.

The above process belongs to wrapping feature selection approaches which build many models with dissimilar subsets of input features and hand-picked those features that have best performance agreeing to the performance metric. Although these approaches are independent of the types of variables, yet they might be computationally expensive.

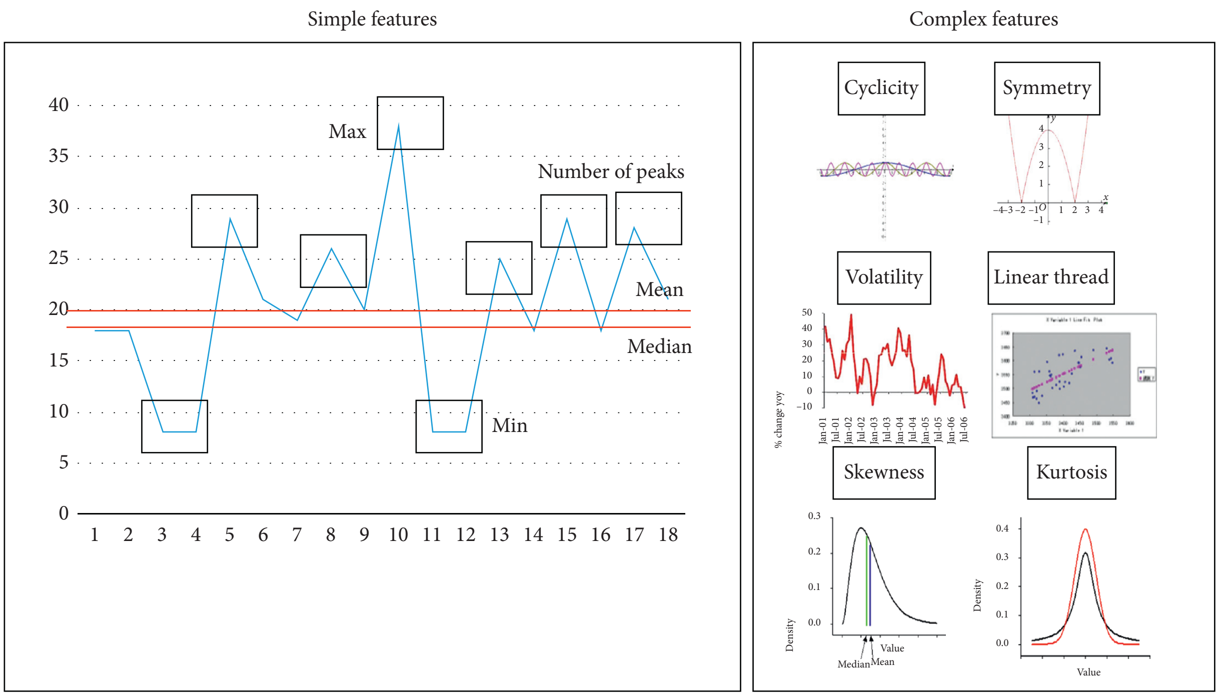


Figure 4: The meaning of features in stream data.

Table 2: The name, design principle, and computing method of some features in Tsfresh.

Name	Design principle	Computing method [16, 30]
Mean	The baseline of time series	$\overline{x} = \text{mean}\,(x_{t-w}\colon x_{t+w})$
Standard deviation	The standard deviation of time series	$\text{std} = \sqrt{\sum_{i=1}^{N}(x_i - \overline{x})^2/N}$
Coefficient of variation	The reflection of the degree of data dispersion	cv = std/mean
Local fluctuation 1	The difference of the smooth curve and original curve	$s_{\text{diff}} = \frac{1}{n}\sum_{i=1}^{n}\lvert x_i - x_i^*\rvert$
Local fluctuation 2	Local fluctuation with a dynamic step	$d\,(\text{step}) = (1/2\text{step})\sum_{i=1}^{n-\text{step}}(x_i - x_{i+\text{step}})^2$
Smooth factor	The ratio of the whole number to the number of turning points	$s_{\text{smooth}} = (1/n-2)N_{\text{change}}$
Symmetrical value	The symmetry of the curve	$\text{sym} = \sum_{i=1}^{n/2} x_i/\sum_{i=n/2}^{n} x_i$
Fluctuation ratio	Whole fluctuation power	$\text{quantil}\,(x_{\text{norm}}, 0.9) - \text{quantil}\,(x_{\text{normal}}, 0.1)$
Skewness	The estimation of the degree of statistical data distribution and the direction of skew is the digital characteristics of the asymmetric degree of statistical data distribution	$S = \left(\sum_{i=1}^{n}(x_i - \mu)\right)^3/n\sigma^3)$
approximate_entropy	Approximate entropy is used to measure the periodicity, unpredictability, and volatility of a time series	Refer to [16]
Autoregressive coefficient	Measure the cyclical nature of data	$\frac{1}{n-1}\sum_{i=1,\ldots,n} 1/(n-l)\sigma^2\sum_{t=1}^{n-l}(X_t - \mu)(X_{t+l} - \mu)$
Kurtosis	The feature number indicating the peak value of the probability density distribution curve at the average value	$E[(X - \mu/\sigma)^4]$
absolute_sum_of_changes	Absolute sum of first-order difference	$\sum_{i=1}^{n-1}\lvert x_{i+1} - x_i\rvert$
Linear_trend	Calculation of a linear least squares regression for the values of the time series to the sequence from 0 to the length of the time series −1	Refer to [16]
fft_aggregated	Returns the variance, mean, kurtosis, skewness, and absolute Fourier transform spectrum	Refer to [16]

Though these features are designed for general classification and clustering problems, and not for algorithm selection problems, as dictated by the literature on machine learning technology, the transform learning technology may perform well in similar problems for various problem fields. Considering the similarity of the above two problems, a conclusion can be drawn that the selected features are useful in the algorithm selection task.

Finally, these features will help choose a suitable algorithm for a certainly given stream of data by training a classification model. As it is mentioned before, if these features could be computed in real time, the decision of

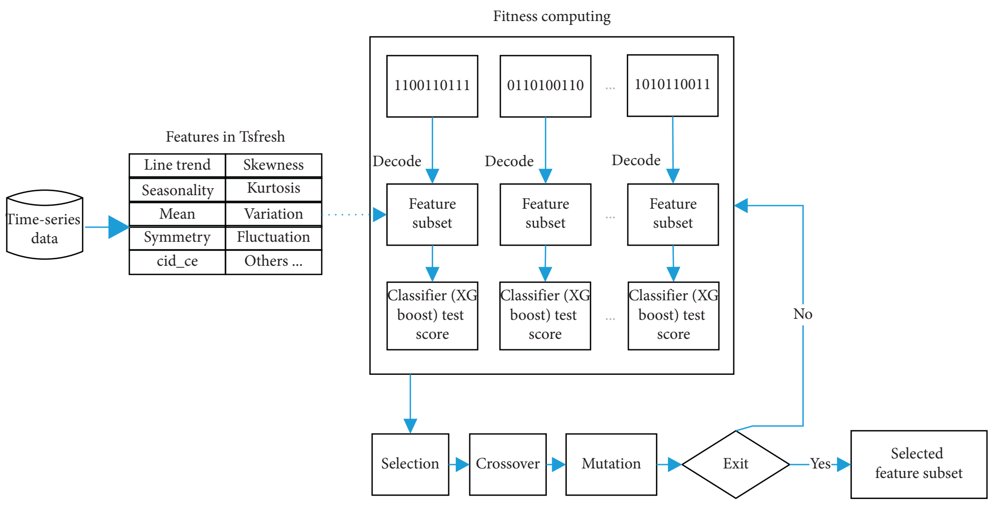


Figure 5: The process of selecting features from Tsfresh features.

choosing the optimum algorithm service will be quicker and thus it will be accepted by many application users.

4. Experimental Validation and Interpretation

4.1. Datasets. For the introduction and assessment of univariate methods, several time-series datasets as listed in Table 3 have been selected. As is presented in Table 3, the reason for selecting these datasets for the assessment of the proposed framework is the availability of similar characteristics in the data. The synthetic and real data encompasses all the commonly known three anomaly forms: random, collective, and point anomaly [14].

4.2. Preprocessing of Data. Standardisation helps numerous machine learning approaches to converge quickly. A dataset is said to be standardised one if its standard deviation σ is 1 and its mean μ is 0. Mathematically, let D be the dataset and σ the standard deviation of D while μ is its mean. Then, standardised D is given by the following equation:

$$\widehat{x} = \frac{x-\mu}{\sigma}, \quad \forall x \in D. \tag{1}$$

4.3. Metrics Evaluation. The performance of our developed framework is evaluated by plotting the receiver operating characteristic (ROC) curve. As a first step, False Positive Rate (FPR) and True Positive Rate (TRP) are illustrated below:

$$\begin{aligned}\text{FPR} &= \frac{FP}{P},\\ \text{TRP} &= \frac{TP}{P},\end{aligned} \tag{2}$$

where FP denotes the total number of wrong positive predictions, TP denotes the total number of correct positive predictions, and P is the total number of positive-labeled values. A list of $\delta \in R$ are used as a threshold that leads to various pairs of FPR and TPR for each δ. A list of two-dimensional coordinates from values already computed is made, and then they will be plotted as a curve. The starting pair of points for this curve will be (0, 0) while the ending pair of points will be (1, 1), respectively. The area under the curve is labeled as AUC. Higher AUC represents the higher possibility that the dignified algorithm allocates anomalous points randomly to the time series. Furthermore, higher anomaly scores than random normal points will enable AUC to correctly associate with various anomaly detection approaches. Thus, in this study, AUC is chosen as an evaluation metric.

4.4. Comparison of Various Methods. Five algorithms out of numerous sets of algorithms such as Long Short-Term Memory Networks (LSTM) [22], Local Outlier Factor (LOF) [25], Prediction Confidence Interval (PCI) [20], One-Class Support Vector Machines (OC-SVM) [26], and Autoencoder [21] are set as baseline algorithms. These algorithms represent machine learning techniques, deep learning techniques, and statistical techniques that are developed for anomaly detection in stream data. Some of the hyperparameters used in our study are borrowed from the work of Bu et al. [14]. Table 4 explains the hyperparameters of these algorithms.

4.5. Experimental Procedure and Outcome Analysis. First, each and every dataset is divided into training set 60% set and testing set 40% using a stratified statistical sampling technique. Each time series of the training dataset and its appropriate algorithm are constructed and computed as a paired dataset. Secondly, the XGBoost model is trained to recognise the patterns of a stream using the paired dataset as an input. Thirdly, the trained XGBoost model is used for the recognition of patterns in each time-series in the test set and finds out a suitable algorithm as a service. Finally, the performance of ADS with the recommended algorithm employed on each time series is evaluated.

The AUC values of the anomaly detection datasets are presented in Table 5. The outcomes presented in Table 5

Table 3: The datasets used in our experiment.

Name	Source	Number of time-series	Number of time stamps	Ratio of anomalous data (%)	Characteristic
Dataset 1	Yahoo [18]	100	1680	0.5	Artificial univariate time-series data comprises of anomalies' change point where it changes the mean of the time series
Dataset 2	Yahoo [18]	100	1680	0.3	Artificial univariate time-series data with anomalies and seasonality are introduced at random points
Dataset 3	Yahoo [18]	100	1421	0.3	Artificial univariate time-series data
Dataset 4	Yahoo [18]	67	1420	1.9	A univariate Yahoo services time-series dataset recording the traffic in which anomalies are by-hand pigeonholed. Majority of the time-series are static
NYCT	NAB [17]	1	10320	0.05	A univariate New York City taxi request time-series dataset comprising the New York City (NYC) taxi demand from July 1, 2014, to January 31, 2015, with an observation of the no. of passengers noted down every half hour. It comprises five shared anomalies that arise in the NYC: Christmas, thanksgiving, marathon, snowstorm, and New Year's Day.

Table 4: Description of our experimental datasets.

Model	Hyperparameter	Value
LOF	Distance function (k)	10, Minkowski distance
PCI	k, α	30, 98.5
LSTM	Filters, optimisers, architecture, loss, batch size, and epochs	$4 * 4$, Adam, 2-state full LSTM layer, MSE, 32, 50
OC-SVM	Upper bound of outliers, kernel	Radial basis function kernel (RBF), 0.7
Autoencoder	Architecture, activation functions, optimiser, loss, batch size, and epochs	Decoding layers (16, 32), encoding layers (32, 16), linear for output, ReLU for encoding and decoding, MSE, Adam, 32, 50

Table 5: The AUC values computed for each experimental dataset using our developed ADSS framework.

Time series	OC-SVM	PCI	LSTM	LOF	Autoencoder	ADSS
Dataset 1	0.939	0.689	0.589	0.952	0.597	0.955
Dataset 2	0.957	0.674	0.578	0.951	0.602	0.956
Dataset 3	0.995	0.762	0.734	0.995	0.743	0.995
Dataset 4	0.851	0.522	0.812	0.814	0.782	0.856
NYCT	0.586	0.54	0.841	0.493	0.697	0.841

proved that for a given dataset the most suitable algorithm may be different in each case. Results presented in Table 5 signpost that LSTM performs best for the NYCT dataset, LOF performs best for dataset 1, while OC-SVM achieves best for datasets 3 and 4. As is given in Table 5, out of the five datasets, our framework shows better performance in four. Even in the case of dataset 2, the performance is nearly equal to the OC-SVM which is the best algorithm. This is the reason; our ADSS framework for algorithm service selection can quickly and flexibly choose the most appropriate algorithm service for any type of data flow processing.

4.6. NAB Dataset and Its Outcome Analysis. In paper [32], researchers compared multiple anomaly detectors such as Skyline, Relative Entropy, and HTM-based algorithms. From its public available experiment reports, we found that Numenta algorithm could achieve the best average performance on all the datasets. However, for one certain dataset such as Twitter_volume_UP, the EarthgeckoSkyline could defeat other detectors. Inspired by the ensemble learning and algorithm selection strategy, we use the supervised learning method to choose a suitable detector for one certain dataset, so we show the experiment on NAB dataset. In NAB results, the evaluation metrics are Standard Score, Reward Low FP rate scores, and Reward low FN rate scores; for more information, one can refer to [17].

The process of the experiment is the same as that explained in Section 5; the performance of the experiments on the NAB dataset is shown in Table 6. As is demonstrated in Table 6, our framework had achieved better performance considering all these detectors as candidate ADAs. A conclusion can be drawn that our framework could recognise the feature of streaming data and help choose a good detector for it and achieve better performance on average.

4.7. Outcome Analysis. In our framework, the algorithm is decided and recommended as best for current stream data and be configured to check the anomalous data. The base algorithms can be added as needed and the available algorithms will become more and more. So, in the long run, when we add enough algorithms to the service pool, the final anomaly detection performance will become better. This framework takes full advantage of metalearning idea which recognises the stream data pattern and configures its best algorithm.

Table 6: The three kinds of scores for the NAB dataset.

Detector	Standard scores	Reward low FP rate scores	Reward low FN rate scores
Numenta HTM using NuPIC v0.5.6∗	70.1	63.1	74.3
Numenta™ HTM∗	64.6	56.7	69.2
htm.core	63.1	58.8	66.2
EarthgeckoSkyline	58.2	46.2	63.9
Relative entropy	54.6	47.6	58.8
Recommended by our framework	**77.9**	**68.1**	**83.2**

Our framework is not creating a new algorithm; instead it is choosing the finest algorithm for any time-series data, thus possibly improving the total performance of the entire IoT system. In general, our framework modifies the quality of service, in the background of encapsulation of algorithm as web service.

5. Conclusion

In practice, it is unfeasible to build a universal method to detect all types of anomalies in IoT stream data; we attempt to discriminate the data pattern and adjust appropriate ADS. Various ADSs can be chosen and then according to their stream data pattern, they can be configured. We attempt to extract features of a stream and select an appropriate algorithm for its anomaly detection.

Experimentations through five datasets (illustrated in Table 3) demonstrate the performance of our method and are presented in Table 5. The experimental outcomes described in Table 5 prove that our method is able to select the accurate service proficiently and can recognise the data pattern efficiently. Moreover, the result on the NAB dataset is shown to further illustrate the good performance achieved by our method.

To further analyse the experimental result, we found that our method is like an ensemble learning process that will merge together different kinds of models in order to achieve better results. Different from the traditional ensemble method, we try to capture the intrinsic characteristics of streaming data from the view of feature engineering. So, the Tsfresh tool and GA algorithm played an important role when finding the importance of features.

However, our method is able to select the service efficiently and can recognise the data pattern efficiently. Still, our method needs sufficient historical data to improve the accuracy of a service selection process that can be done by collecting further real-world data and experimenting with more artificial dataset in the future.

Data Availability

All the data used to report the findings in this paper are provided in the form of tables in the paper.

Conflicts of Interest

There are no conflicts of interest as declared by the authors of this paper.

Acknowledgments

This work was supported by Projects of International Cooperation and Exchanges NSFC (Grant no. 62061136006) and the Scientific Research Initiation Funds (Grant nos. 2462020YJRC001 and 110051360002).

References

[1] V. Chandola, V. Mithal, and V. Kumar, "Comparative evaluation of anomaly detection techniques for sequence data," in *Proceedings of Eighth IEEE International Conference on Data Mining*, pp. 743–748, Pisa, Italy, December 2008.

[2] P. Banerjee, R. Friedrich, C. Bash et al., "Everything as a service: powering the new information economy," *Computer*, vol. 44, no. 3, pp. 36–43, 2011.

[3] Q. Chen, M. Hsu, and H. Zeller, "Experience in continuous analytics as a service (CaaaS)," in *Proceedings of International Conference on Extending Database Technology*, ACM, Uppsala, Sweden, March 2011.

[4] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by internet of things," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 1, pp. 81–93, 2014.

[5] Z. H. Ali, H. A. Ali, and M. M. Badawy, *A New Proposed the Internet of Things (IoT) Virtualization Framework Based on Sensor-As-A-Service Concept*, Wireless Personal Communications, Berlin, Germany, 2017.

[6] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.

[7] M. Shafiq, Z. Tian, A. K. Bashir et al., "Data mining and machine learning methods for sustainable smart cities traffic classification: a survey," *Sustainable Cities and Society*, vol. 60, 2020.

[8] Y. Han, C. Liu, S. Su, M. Zhu, Z. Zhang, and S. Zhang, "A proactive service model facilitating stream data fusion and correlation," *International Journal of Web Services Research*, vol. 14, no. 3, pp. 1–16, 2017.

[9] H. Ren, B. Xu, Y. Wang et al., "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3009–3017, Anchorage, AK, USA, August 2019.

[10] L. Stojanovic, M. Dinic, N. Stojanovic et al., "Big-data-driven anomaly detection in industry (4.0): an approach and a case study," in *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data)*, IEEE, Washington, DC, USA, December 2016.

[11] Z. Zhang, J. Yu, X. Li et al., "A data-driven service creation approach for effectively capturing events from multiple sensor streams," in *Proceedings of the 2019 IEEE International*

Conference on Web Services (ICWS), pp. 346–354, IEEE, Milan, Italy, July 2019.

[12] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[13] M. Braei and S. Wagner, "Anomaly detection in univariate time-series: a survey on the state-of-the-art," Preprint, 2020.

[14] J. Bu, Y. Liu, S. Zhang et al., "Rapid deployment of anomaly detection models for large number of emerging KPI streams," in *Proceedings of 2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, pp. 1–8, IEEE, Orlando, FL, USA, November 2018.

[15] Z. Yang, W. Ding, Z. Zhang, H. Li, M. Zhang, and C. Liu, "A service selection framework for anomaly detection in IoT stream data," in *Proceedings of 2020 International Conference on Service Science*, pp. 155–161, ICSS, Xining, China, August 2020.

[16] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–A python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018.

[17] A. Lavin and S. Ahmad, "Evaluating real-time anomaly detection algorithms-the Numenta anomaly benchmark," in *Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications*, pp. 38–44, (ICMLA), Miami, FL, USA, December 2015.

[18] Yahoo! Webscope Dataset Ydata-Labeled-Time-Series-Anomalies-V10 2010, http://labs.yahoo.com/Academic_Relations.

[19] E. Ostertagova and O. Ostertag, *The Simple Exponential Smoothing Model*, Monash University, Melbourne, UK, 2011.

[20] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[21] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of ACM International Conference Proceeding Series*, pp. 4–11, New York, NY, USA, December 2014.

[22] P. Malhotra, A. Ramakrishnan, G. Anand et al., "LSTM-based encoder-decoder for multi-sensor anomaly detection," arXiv Prepr arXiv160700148, 2016.

[23] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial Databases with Noise," in *Proceedings of KDD-96*, Munchen, Germany, 1996.

[24] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 1–39.

[25] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 93–104, 2000.

[26] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proceedings of the International Joint Conference on Neutral Networks*, Los Alamos, New Mexico, July 2003.

[27] J.-B. Kao and J.-R. Jiang, "Anomaly detection for univariate time series with statistics and deep learning," in *Proceedings of 2019 IEEE Eurasia Conference on IOT, Communication and Engineering*, pp. 404–407, ECICE), Yunlin, Taiwan, October 2019.

[28] A. Bagozi, D. Bianchini, V. De Antonellis, M. Garda, and A. Marini, "A relevance-based approach for big data exploration," *Future Generation Computer Systems*, vol. 101, pp. 51–69, 2019.

[29] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system," in *Proceedings of Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco, CF, USA, August 2016.

[30] N. Laptev, S. Amizadeh, and I. Flint, "Generic and scalable framework for automated time-series anomaly detection," in *Proceedings of Knowledge Discovery and Data Mining*, pp. 1939–1947, Sydney, UK, August 2015.

[31] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, 1998.

[32] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.