IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 10, 2021, accepted February 19, 2021, date of publication March 2, 2021, date of current version March 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3063671

Research on Intrusion Detection Based on Particle Swarm Optimization in IoT

JINGYU LIU[1,2,3], DONGSHENG YANG[2,3], MENGJIA LIAN[1,2,3], AND MINGSHI LI[1,2,3]

[1]School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China
[2]Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China
[3]Liaoning Key Laboratory of Domestic Industrial Control Platform Technology on Basic Hardware & Software

Corresponding author: Dongsheng Yang (dsyang@sict.ac.cn)

This work was supported by the Research on Key Technologies of Real-time Fault Diagnosis in Intelligent Production System based on Industrial IoT under Grant No. 2017YFE0125300.

ABSTRACT With the advent of the ''Internet plus'' era, the Internet of Things (IoT) is gradually penetrating into various fields, and the scale of its equipment is also showing an explosive growth trend. The age of the ''Internet of Everything'' is coming. The integration and diversification of IoT terminals and applications make IoT more vulnerable to various intrusion attacks. Therefore, it is particularly important to design an intrusion detection model that guarantees the security, integrity and reliability of the IoT. Traditional intrusion detection technology has the disadvantages of low detection rate and poor scalability, which cannot adapt to the complex and changeable IoT environment. In this paper, we propose a particle swarm optimization-based gradient descent (PSO-LightGBM) for the intrusion detection. In this method, PSO-LightGBM is used to extract the features of the data and inputs it into one-class SVM (OCSVM) to discover and identify malicious data. The UNSW-NB15 dataset is applied to verify the intrusion detection model. The experimental results show that the model we propose is very robust in detecting either normal or various malicious data, especially small sample data such as Backdoor, Shellcode and Worms.

INDEX TERMS Intrusion detection, Internet of Things (IoT), particle swarm optimization (PSO), one-class SVM (OCSVM).

I. INTRODUCTION

The Internet of things (IoT), regarded as the next generation interconnection mode, is a ubiquitous network built on the Internet [1]. In the age of digital transformation, it has become a well-known technology, which connects a large number of micro devices (smart phones, electricity meters, etc.) with the Internet according to the agreed protocol, achieving the intelligent communication between people, people and things, and things and things. It is the high-tech behind smart city, self-driving and intelligent residence. With the gradual scale of the IoT, its industrial value will be 30 times larger than the Internet, and it will become the next trillion level information industry business. According to the flow of information collection, transmission and processing, the hierarchical structure of IoT can be divided into 3 layers from bottom to top [2], as shown in Figure 1.

A large number of traditional equipments in digital transformation, almost have no synchronous configuration of protection capabilities, affecting the integrity, security and reliability of the IoT. At the same time, due to the integration and diversification of IoT terminals and applications, it brings more security uncertainty to IoT business. The ever-growing variety of IoT interconnected devices provide attackers with a huge and extensive network intrusion portal, leading to the IoT facing huge security problems. Today, the intrusion attack problems faced by the IoT are mainly distributed in the three-layer structure, namely the perception layer, the network layer and the application layer, as shown in Figure 2. People usually use data confidentiality, data integrity, data privacy, identity authentication, authorization and access control, intrusion detection and other technologies to develop corresponding preventive measures to solve the intrusion problems of the IoT [3].

Intrusion detection system (IDS), proposed by Anderson [4], can discover the intrusion behavior of terminal equipment and IoT communication process in the IoT environment, and then take corresponding defense measures according to the detected behavior. In the field of intrusion detection, IDS can be divided into different dimensions,

The associate editor coordinating the review of this manuscript and approving it for publication was Tyson Brooks.

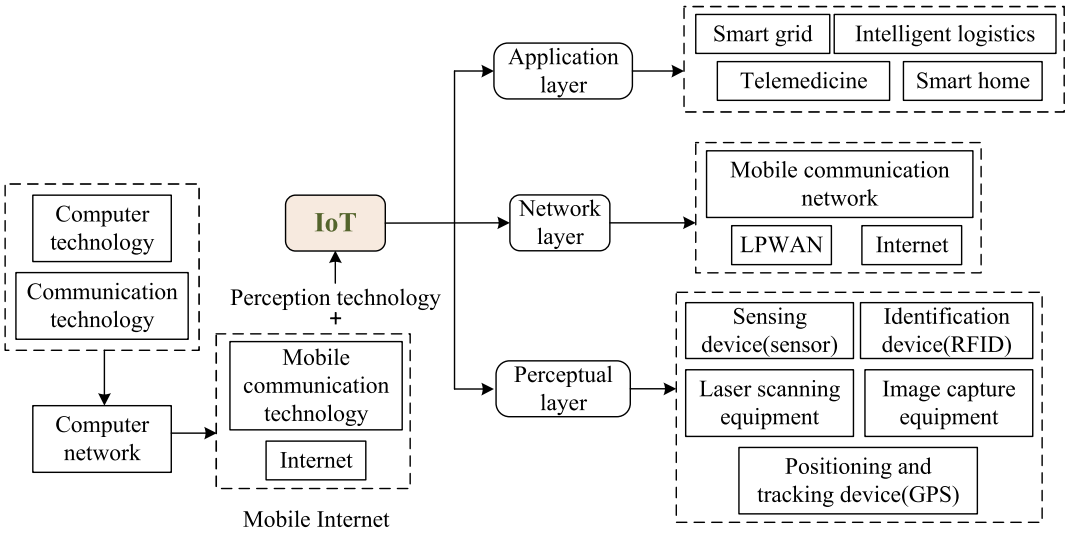


FIGURE 1. Typical IoT architecture.

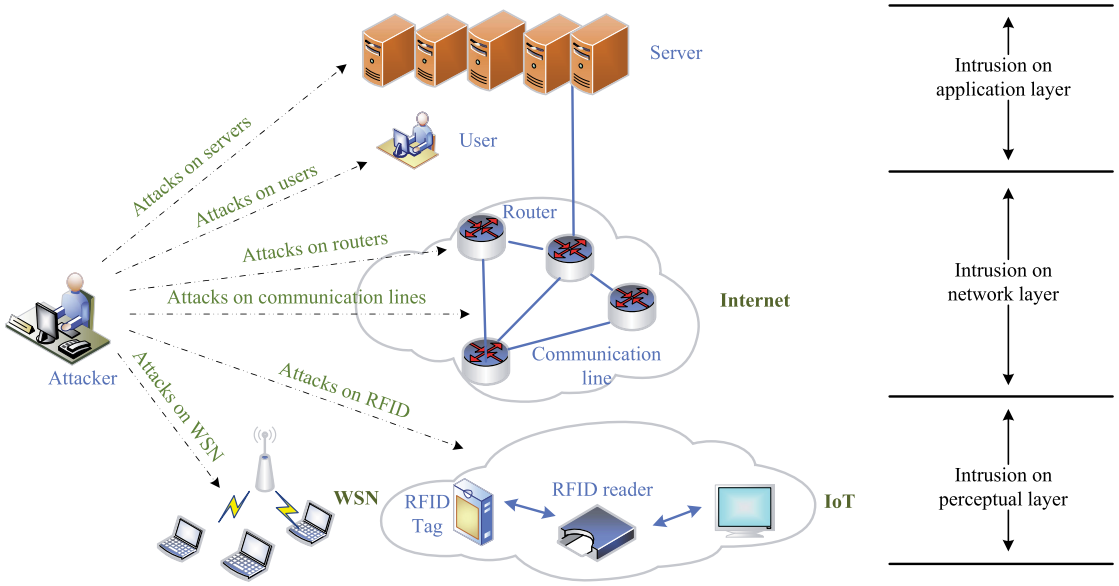


FIGURE 2. Types of attacks faced by different layers of the IoT.

as shown in Figure 3. In the light of the detection technology used by the system, IDS can be separated into anomaly-based IDS (AIDS) and misused-based IDS (MIDS). AIDS has the advantage of detecting new types of intrusions. It establishes a profile of normal activities through machine learning. When the current subject's activities deviate from the normal profile, it is considered that it may be an "intrusion" behavior generated by different mechanisms [5]. Based on the known system defects and intrusion patterns, MIDS can find some characteristic attacks accurately, but it relies too much on the pre-defined security policies to identify unknown attack behaviors of the system, resulting in missed detection [6]. MIDS has higher accuracy and lower false alarm rate (FAR) than AIDS [7], however, the latter has the ability to reveal unknown attacks [8], so it has better scalability and practicability than MIDS [9]. In terms of data sources, IDS can be split into host-based IDS (HIDS) and network-based IDS (NIDS). HIDS, which is efficient, detects intrusions by monitoring and analyzing the audit records of hosts, nevertheless, has poor execution. NIDS is mainly deployed on network nodes, which is capable of listening to and collecting data on the shared network segment in real time, so as to analyze suspicious phenomena. In accordance with the working mode, there are two IDSs, offline IDS and

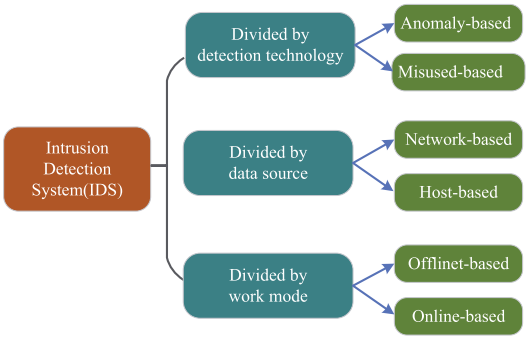


FIGURE 3. Classification of IDS.

online IDS. The former, a non real-time system, analyzes audit events after the event and checks the intrusion activities, while the latter is a real-time online detection system, which includes real-time network packet analysis and real-time host audit analysis.

With the development of IoT, the traditional first security protection technologies, such as identity authentication, firewall and data encryption can no longer cope with the constantly updated intrusion attacks [10]. Therefore, in order to effectively detect the attacks of IoT, NIDS, as the second protection technology, is supposed to be deployed in IoT [11]. In addition to NIDS, due to the rapid update of IoT devices and the mass transmission of data, the entire IoT system is faced with many unknown types of attacks, so AIDS has become another barrier to protect the security of IoT. In recent years, a large number of IDSs based on data mining [12], machine learning [13], rules and statistics-based models [14], [15], neural networks [16] and other related algorithms have been established, whereas these methods are difficult to adapt to fast-developing attacks, which have a high false alarm rate and cannot achieve accurate detection of new types of attacks.

Hindy *et al.* [17] summarized the dataset for evaluating IDSs from 2008 to 2020. Among a total of 85 papers on IDSs, 96.93% (the highest proportion) of the papers used machine learning methods in building IDSs, of which 14.20% (the second highest proportion) applied SVM-based intrusion detection methods. Moreover, in reality, there are many kinds of attack data, and the number is less than the normal data, and the traditional SVM is not robust to the detection and recognition of multi class data and low frequency data, so we innovatively use OCSVM to complete the detection and recognition of data.

In this study, we use the particle swarm optimization-based model to deploy a new type of AIDS to IoT, so that it has higher detection rate and lower false alarm rate in the detection of unknown attacks. Particle swarm optimization (PSO) is an intelligent bionic algorithm, mainly used to solve complex optimization problems, which owns the characteristics of fast convergence and high search accuracy [18]. First of all, this study introduces the LightGBM algorithm on the basis of PSO to achieve double dimensionality reduction and feature extraction of data. Secondly, we use one-class SVM (OCSVM) to model the feature-processed data, and then detect and identify normal and various abnormal data. The proposed model is verified by UNSW-NB15 [6], [19] dataset. The experimental results show that this technology performs better in the aspects of accuracy, detection rate, false alarm rate and time cost, compared with other intrusion detection technologies, which proves the possibility of its deployment in real-time IoT environment.

The main contributions of this paper are as follows:

1) The UNSW-NB15 dataset has an extremely unbalanced distribution of normal and various attack data. The application of LightGBM on PSO can effectively avoid the problem of uneven distribution of large-scale datasets.
2) PSO-LightGBM is used to double dimension reduction and feature extraction, which greatly reduces the size of the dataset.
3) We apply OCSVM to build a model on various data to test the effectiveness, implementability and robustness of the proposed method.

The structure of this paper is as follows. In section II, we introduce in detail the technologies and its principles used in this research. Section III is an overview of IoT-based IDS and intrusion detection technologies. Section IV is the core part of this article, which presents the framework and implementation of the proposed model. Section V is a comparison and analysis of the experimental results. Finally, the summary and prospect are in Section VI.

II. BACKGROUND

A. PARTICAL SWARM OPTIMIZATION (PSO)

Particle swarm optimization (PSO), an intelligent bionic algorithm, originated from the study of bird predation behavior, was proposed by Kennedy and Eberhart in 1995 [20]. The basic idea of PSO is to find the optimal solution through the cooperation and information sharing among individuals in the group. Specifically, the position and speed of each bird are independent variables, and the food density of each arriving place is a function value. Each search will adjust its search direction and speed according to the difference between the optimal location of its own history search and that of the population history search. In the end, the whole bird swarm can gather around the optimal location of the population, thus finding the optimal solution, that is, the problem convergence. The main advantages of PSO are as follows:

1) Compared with the traditional algorithm, PSO has a very fast computing speed and strong global search ability [18].
2) PSO is not sensitive to population size, which has little effect on training speed.
3) When optimizing the objective function, there is no need to calculate the gradient information of the function, and there are no restrictions for continuity, derivability, convexity, and connectivity of feasible regions for the objective function [21].

Researchers usually abstract PSO as a random search problem in D-dimensional space, with the goal of optimizing the objective function. In the D-dimensional space, n particles constitute population $p_i = (p_{i1}, p_{i2}, \ldots, p_{iD})^T$, and the i-th particle consists of a d-dimensional position vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})^T$ and a velocity vector $v_i = (v_{i1}, v_{i2}, \ldots, v_{id})^T$. For each particle in the population, fitness value can be obtained according to fitness function to evaluate the fitness of particles [22]. This paper adopts the fitness function proposed by Vieira [23], as shown in (1), where α is a hyperparameter, which can coordinate the relationship between the classifier performance P (including accuracy, F-score, precision, etc.) and the size of the feature subset N_f relative to the total number of features N_t.

$$F(X) = \alpha(1 - P) + (1 - \alpha)\left(1 - \frac{N_f}{N_t}\right) \tag{1}$$

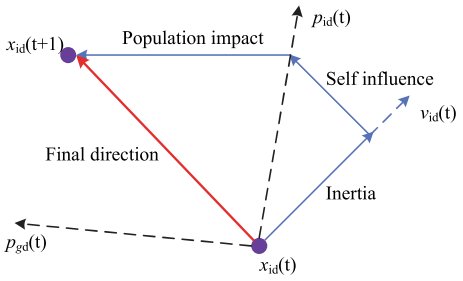


FIGURE 4. Update of position and velocity of particles.

When the particle i searches in the D-dimensional space, it is initialized to a group of random particles, and the optimal solution is found through iteration. With the continuous search of the particles, the optimal position $p_i = (p_{i1}, p_{i2}, \ldots, p_{id})^T$ found by itself is the local optimal solution, and the velocity is denoted as $v_i = (v_{i1}, v_{i2}, \ldots, v_{id})^T$. The optimal position $p_g = \left(p_{g1}, p_{g2}, \ldots, p_{gd}\right)^T$ found by the entire particle swarm is the global optimal solution. In each iteration, the particle updates its position and velocity by tracking two "optimal solutions", that is, $\left(p_i, p_g\right)$, as shown in Figure 4. The formula used for the update is shown in (2) (3):

$$\begin{aligned} v_{id}(t+1) &= \omega v_{id}(t) + c_1 r_1 \left(p_{id}(t) - x_{id}(t)\right) \\ &\quad + c_2 r_2 \left(p_{gd}(t) - x_{id}(t)\right), \end{aligned} \tag{2}$$

$$\begin{aligned} x_{id}(t+1) &= x(t) + v_{id}(t+1), \\ &\quad i = 1, 2, \ldots, N; d = 1, 2, \ldots, D, \end{aligned} \tag{3}$$

where N is the total number of particles in the population; d shows the d-th dimension of particle i; t is the number of current iterations; ω reveals a non-negative inertia factor that regulates the global and local optimization capabilities. The larger the value, the stronger the global optimization ability and the local optimization ability is weaker, and vice versa; $v_{id}(t)$ and $v_{id}(t+1)$ represents the current and the updated velocity of the particle, respectively; c_1 and c_2 are acceleration factors, usually $c_1 = c_2 = 2$, where r_1 and r_2 are two random numbers with values of (0, 1) [24]. In order to increase the randomness of particle search and prevent blind search, the position and velocity of the particles are limited to $[-x_{\max}, x_{\max}]$ and $[-v_{\max}, v_{\max}]$. PSO algorithm is outlined in Algorithm 1.

Algorithm 1 Particle Swarm Optimization (PSO)

Input: N: population size
p_i: local optimal position
p_g: group optimal position
fit: fitness function
Output: p_g

```
randomly initialize the position x_i and velocity v_i of particle i
while criterion is not met do
    for i=1 to N do
        calculate the fitness value of each particle according to the fitness function
        if fit(x_i) is greater than fit(p_i) then
            p_i ← x_i
        if fit(p_i) is greater than fit(p_g) then
            p_g ← p_i
        update the position and velocity of particle i
return p_g
```

B. LightGBM

Light gradient boosting machine (LightGBM) is an integrated algorithm for building gradient boosting decision tree (GBDT), which has the characteristics of faster training speed, lower memory consumption, better accuracy, and support for parallel processing of massive data [25]. The proposal of LightGBM solves the problems encountered by GBDT in processing massive data, so that GBDT can be applied to practice better and faster. Different from traditional algorithms for generating GBDTs, such as XGBoost [26], pGBRT [27], scikit-learn [28], etc., LightGBM mainly optimizes the following aspects:

1) GRADIENT-BASED ONE-SIDE SAMPLING (GOSS)

GOSS mainly realizes data sampling. Since large gradient samples have a greater impact on information gain, GOSS discards samples which are not helpful in calculating information gain. When data sampling is performed, only large gradient instances are retained, and small gradient instances are randomly sampled while introducing constant multipliers $\frac{(1-a)}{b}$, leading to make the algorithm pay more attention to the instances of insufficient training and reduce the impact on the distribution of the original dataset. The specific algorithm process is shown in Figure 5, where the parameter d is the number of iterations; a and b are the sampling rates of large and small gradient data, respectively.

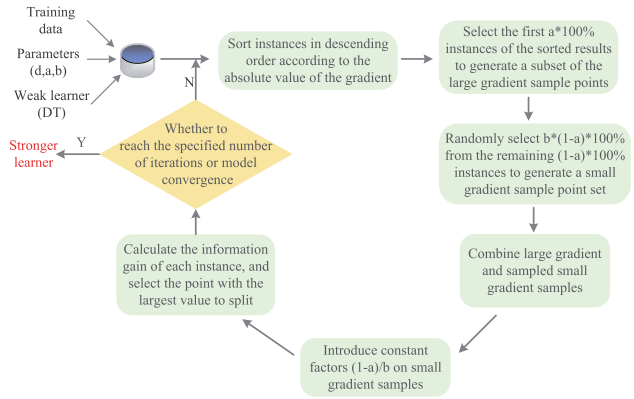


FIGURE 5. GOSS algorithm.

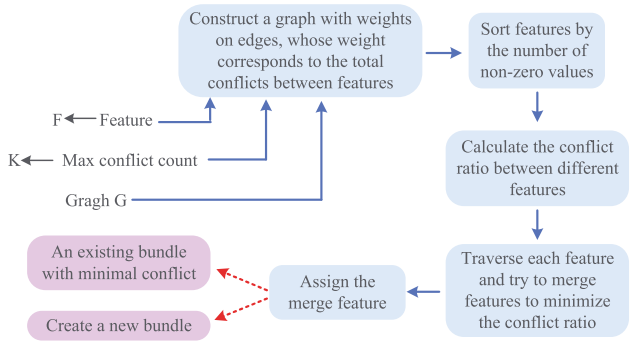


FIGURE 6. EFB algorithm.

2) EXCLUSIVE FEATURE BUNDLING (EFB)

EFB is mainly used for data sampling. Because high-dimensional data is usually sparse, and many features are mutually exclusive in the sparse feature space, EFB fuses these mutually exclusive features, reducing the bundling problem to graph coloring problem to cut down the number of features. If the two features are not completely mutually exclusive, we use the conflict ratio to measure the degree of non-mutual exclusion between features. When the value is small, we can choose to bundle the two features which are not completely mutually exclusive without affecting the final accuracy. Figure 6 shows the algorithm process of EFB, whose time complexity is $O\left(\#feature^2\right)$, and it can be processed only once before model training.

3) HISTOGRAM-BASED ALGORITHM

Firstly, the number of bins needed by each continuous float feature is determined, as well as an integer is assigned to each bin. Then, the range of float point number is split into k intervals, making the number of bins equal to the number of intervals. After traversing the data once, we can search for the optimal point in accordance with the statistics accumulated in the histogram. In view of the discretization operation of floating-point numbers, the time complexity of the algorithm drops from $O\left(\#data * \#features\right)$ to $O\left(k * \#features\right)$, $k \ll \#data$.

4) LEAF-WISE STRATEGY

On the basis of histogram algorithm, LightGBM does not exploit level-wise strategy, which is commonly used in GBDT framework, as the growth approach of decision tree. Instead, it applies leaf-wise strategy with depth restrictions, selecting the node with the largest splitting gain from all current leaf nodes to split until the iteration is completed. In addition to the leaf-wise tree growth strategy, another optimization of LightGBM is that the histogram of leaf node can be obtained by the difference between the histogram of father node and that of sibling node. See Figure 7 for details.

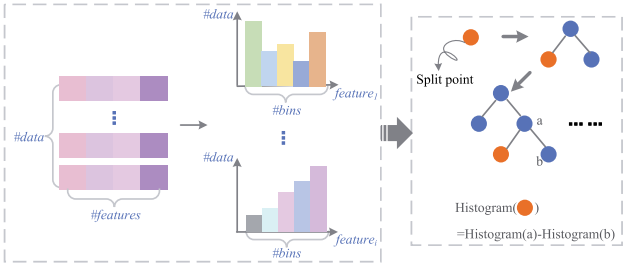


FIGURE 7. Histogram-based algorithm and leaf-wise strategy.

C. ONE-CLASS SVM (OCSVM)

In the training phase, OCSVM has only one type of data (positive or negative), which tries to separate all data points from the zero point in the feature space F, and maximizes the distance between the hyperplane and the zero point [29]. This algorithm usually utilizes a sphere instead of a plane, minimizing the volume of the hypersphere by obtaining the spherical boundary of the data in a high-dimensional space, thereby minimizing the influence of outliers [30], as shown in Figure 8.

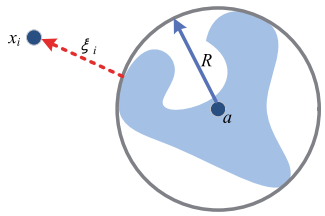


FIGURE 8. OCSVM.

Define a hypersphere $f(x;\omega)$ with closed data, whose center is a and radius is R. In addition, a slack variable ξ_i ($\xi_i > 0, \forall i$) with penalty coefficient C is introduced to make the model more robust, so the optimization problem of OCSVM is shown in (4).

$$\min_{R,a}\left(R^2 + C\sum_{i=1}^{n}\xi_i\right)$$
$$s.t.\ \ ||x_i - a||^2 \le R^2 + \xi_i, \xi_i > 0, \quad i = 1, 2, \ldots, n \qquad (4)$$

III. RELATED WORK

This section mainly introduces the IDS based on IoT and various algorithms used in IDS in recent years, focusing on the analysis of IDS deployed on IoT and IDS based on machine learning.

A. IoT-BASED IDS

Hosseinpour *et al.* [31] proposed an intrusion detection system based on artificial immunity. The system included three layers: cloud, fog and edge. In the cloud layer, IDS gathered important network traffic while training the detector of the IDS. Intelligent data was used to analyze intrusion alerts, and finally detectors were deployed at the edge layer, while this detection system needed a complicated network structure to realize, making it difficult to deploy.

Nobakht *et al.* [32] have studied a host-based lightweight IDS. Its core IDS ran on the application layer nodes of the IoT, and the perception layer nodes were only responsible for collecting data, so the system was only applicable to small IoT systems.

Alotaibi *et al.* [33] established a voting-based IDS. For each time period's node in the IoT system, one of the three pre-trained intrusion detection algorithms was selected for active intrusion detection. Different intrusion detection algorithms could avoid the problem of various detection rates for different attacks by increasing randomness.

Pajouh *et al.* [34] proposed an intrusion detection algorithm with dual dimensionality reduction and dual detection, which was applied to each node in the same layer of the IoT to realize the identification of malicious attacks or behaviors, speeding up the training by reducing the feature dimension of nodes. Although the node has been detected twice, it still lost a certain accuracy, and the detection rate of attack types with a small number of samples was low.

Focusing on the TCP/IP model, Moustafa *et al.* [35] extracted the flow indicators needed to construct NIDS from MQTT, DNS and HTTP protocols, evaluating the indicators by correlation entropy and correlation coefficient. Finally, the IoT anomaly detection was realized by adaboost assemble method integrating decision tree (DT), naive bayes (NB) and artificial neural network (ANN). Because of various protocols and diverse characteristics of different protocols in IoT, it was hard to construct dynamic configuration files of normal and attack patterns.

Roux *et al.* [36] utilized neural networks to monitor and detect wireless communications to protect interconnected smart homes and smart factories in IoT. However, the proposal had not yet defined the actual IoT network configuration and lacked effective experimental evaluation.

B. INTRUSION DETECTION TECHNOLOGY

Tian *et al.* [37] firstly preprocessed the data features with probabilistic mass function (PMF) and min-max normalization, and then improved the likelihood function of DBN in the unsupervised training stage, that is, introducing the penalty term combination based on kullback leibler (KL) divergence and non-mean Gaussian distribution to resolve the feature homogenization and over fitting.

Benmessahel *et al.* [38] combined artificial neural network (ANN) and multiverse optimizer (MVO), mainly exploited MVO to train feed-forward multilayer artificial neural networks to form an evolutionary neural network (ENN) technology for detecting new types of attacks. However, this method did not perform feature selection on the dataset, so it was not suitable for processing large datasets.

Ramp loss k-support vector classification-regression (Ramp-KSVCR), a multi-class anomaly detection method, proposed by Bamakana *et al.* [39]. Firstly, the K-SVCR model was used to deal with the attack distribution's extreme imbalance and skewness, furthermore, the ramp loss function was applied to reduce the sensitivity of the SVM model to noises and outliers in the training phase, and finally the concave-convex procedure (CCCP) was introduced to work out the non-differentiable and non-convex problem of the Ramp-KSVCR model. Although it solved the three main problems of data, distribution imbalance, noise interference and scalability in intrusion detection, it was too hasty in the data preprocessing stage, and did not take the feature selection and filtering of data into account. Moreover, the model needed to be retrained to build a model when dealing with new data, which greatly limited its scalability.

Reference [40] used the unsupervised deep auto-encoder (DAE) algorithm for continuous training, learning normal behaviors while generating optimal parameters, inputting them into the supervised deep neural network, which could effectively adjust the parameters of the neural network and classify the network data, but it was not sensitive to the information transmission between different protocols.

Reference [41] combined incremental extreme learning machine (I-ELM) with adaptive principle component analysis (A-PCA) to form a new IDS. A-PCA could automatically extract the characteristics of the network number effectively under parameter constraints, while I-ELM was used for detecting malicious attacks. This paper exploited the UNSW-NB15 dataset to experiment and verify the proposed IDS. Its accuracy, detection rate and false alarm rate were 70.51%, 77.36% and 35.09%, respectively, which still had very large development space.

Reference [42] proposed an IDS based on genetic algorithm and LSSVM for feature selection. In this paper, three parameters including true positive rate (TPR), false positive rate (FPR) and the number of selected features were used as the fitness function of genetic algorithm for iteration. After the iteration stop criterion was reached, LSSVM got used to train the model and classify the data. The author applied the UNSW-NB15 dataset to prove the effectiveness of this method, while it did not involve the detection of the four types of low-frequency data: Analysis, Backdoor, Shellcode and Worms, so the feasibility of the IDS could not be fully demonstrated.

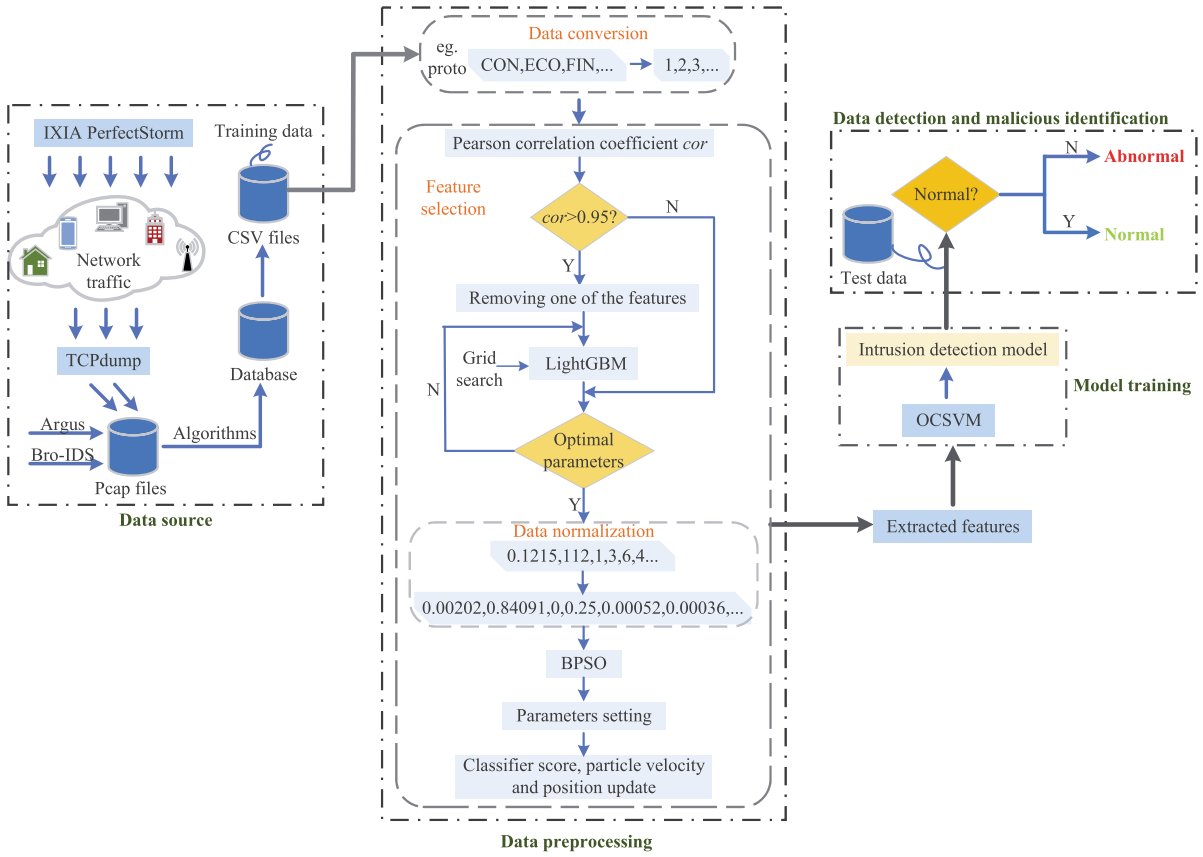


FIGURE 9. Proposed intrusion detection model.

Reference [43] presented NIDS for large-scale unbalanced datasets, designing a class-unbalanced technology, SGM, which used SMOTE oversampling and GMM-based gathering sampling technology to achieve the same amount of various data, and then imported the data into convolutional neural network for data classification. This model not only excelled in detection rate, precision and F1 score, but also outperformed the current existing IDS model in terms of training and detection time. However, when the researchers made use of the method in [44] to perform feature reduction on the UNSW-NB15 dataset after one-hot encoding, the number of features has changed from 208 to 202, which was only 6 fewer. If a more effective special diagnosis extraction method is used, the NIDS will be more perfect.

From section III-A and section III-B, it can be seen that today's intrusion detection technologies mainly focus on three problems: unbalanced dataset distribution, parameter optimization and feature extraction, and whether the intrusion detection algorithm is efficient or not directly determines the overall performance of IDS. The PSO-LightGBM feature extraction method proposed in this paper uses grid search to optimize parameters on the basis of default parameters, which makes the content of dataset more refined and effective. In the aspect of detection, considering the unbalanced distribution of all kinds of data in UNSW-NB15 dataset, we utilize OCSVM to model the data after feature extraction, ensuring the detection rate and accuracy of the data, and the whole process does not cost a lot of time. See section IV for details.

IV. FRAMEWORK ARCHITECTURE

In this section, we show the proposed IDS, which mainly includes three parts: data acquisition, data preprocessing, and model building. Firstly, we perform feature extraction on the UNSW-NB15 dataset, and then apply the processed data for model training and use it to detect and identify the abnormal behavior of the data. The detailed process is shown in Figure 9.

A. DATA SOURCE

The UNSW-NB15 dataset used in this research was collected by ACCS Lab with the help of IXIA PerfectStorm tool to display the network flow dataset where normal data and malicious attacks coexisted in a real network environment. Using Argus and Bro-IDS tools, plus 12 algorithms, the researchers generated 49 labelled features for each record. These features can be parted into seven categories: flow features, base features, content features, time features, general purpose features, connection features and labeled features, furthermore, it is about 3 binary features, 2 timestamp features,

6 nominal features, 10 float features and 28 integer features. In addition to Normal data, it also includes nine types of attacks, namely, Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode and Worms. This paper uses the "UNSW_NB 15_training-set.csv" as training set (175341 records) and "UNSW_NB15_testing-set.csv" as test set (82332 records) with 43 features configured by the R & D staff to conduct experiments [6].

B. DATA PREPROCESSING

Accuracy, integrity and consistency are three important factors to ensure the quality of data. However, a large number of incorrect, incomplete and inconsistent data exist in large databases and data warehouses in the real world. In the face of so much raw data, data preprocessing can filter out redundant data irrelevant to modeling, which is able to not only reduce the dimension of data, but also accelerate the training speed of model. In this study, data preprocessing mainly includes the following four steps.

1) DATA CONVERSION

Since the model constructed in this paper requires the input of numerical data, it is necessary to convert the qualitative data in the original dataset into numerical data. Taking the "state" feature of UNSW-NB15 dataset as an example, the attribute of "state" is replaced by a unique integer, that is, CON = 1, ECO = 2, FIN = 3, ...

2) FEATURE EXTRACTION

Feature extraction is the process of reducing the considered feature variables by obtaining a set of main variables that are basically important features. It can save storage space, accelerate calculation speed, remove redundant features, reduce noise, and avoid model overfitting.

1) Pearson correlation coefficient. Pearson correlation coefficient [45] is defined as the quotient of covariance and standard deviation between two variables, which describes the closeness of the relationship between two spaced variables. Its value is between $[-1, 1]$, which is denoted as *cor*, and the formula is shown in (5).

$$cor = \frac{COV(X, Y)}{\sigma_X \sigma_Y} = \frac{E((X - \mu_X)(Y - \mu_Y))}{\sigma_X \sigma_Y}, \quad (5)$$

The degree of linear correlation between variables is proportional to *cor*. When *cor*=0, it indicates that the two variables are not linear, but there may be other ways of correlation, for example, curve correlation. Two features with a larger *cor* coexisting in the model will reduce the robustness and generalization of the model, therefore, for the features with $cor > 0.95$, we select one to delete, where X and Y represent two random variables, μ and σ represent the mean and variance of corresponding random variables, respectively. After calculation, there are a total of 12 sets of features $cor > 0.95$. From Table 1, there are 9 features, namely, "sbytes", "dbytes", "sloss", "dloss", "dwin","ct _src_dport_ltm", "ct_dst_src_ltm", "ct_ftp_cmd" and "ct _srv_dst" will be dropped. The heatmap between these strongly related features is shown in Figure 10.

TABLE 1. Strong correlation features and corresponding *cor*.

Count	Feature X (droppped)	Feature Y	*cor*
1	dbytes	dpkts	0.956099
2	ct_src_dport_ltm	ct_dst_ltm	0.962126
3	dloss	dpkts	0.963805
4	ct_dst_src_ltm	ct_srv_src	0.968708
5	ct_srv_dst	ct_dst_src_ltm	0.973690
6	sbytes	spkts	0.975092
7	sloss	spkts	0.980218
8	ct_srv_dst	ct_srv_src	0.981081
9	dwin	swin	0.989966
10	dloss	dbytes	0.995618
11	sloss	sbytes	0.997076
12	ct_ftp_cmd	is_ftp_login	1.000000

FIGURE 10. Heatmap between strongly correlated features.

2) LightGBM. Section II-B mentioned that LightGBM is an algorithm for quickly establishing GBDT. It contains many parameters while different parameter settings have distinct effects on the model. The following focuses on several parameters that have a greater impact on LightGBM.

- learning_rate, the learning rate of the model.
- max_depth, which limits the maximum depth of the tree and can also control the complexity of the leaf nodes. When the amount of data is small, overfitting can be prevented, so that the tree can still follow leaf-wise growth. But max_depth is not responsible for controlling the depth in LightGBM, so it can be set to the default value, that is -1.
- num_leaves, the number of leaf nodes on the spanning tree. Since LightGBM builds generate trees with a leaf-wise strategy, this parameter can be used to adjust the tree's complexity.
- max_bin, the maximum histogram interval. The smaller the value, the smaller the width of histogram, the node segmentation is more accurate, making the model training faster.

TABLE 2. The value of the parameter in LightGBM.

Parameter	Value	Parameter	Value
n_estimators	98	min_split_gain	0.1
learning_rate	0.01	bagging_fraction	0.9
max_depth	7	bagging_freq	60
num_leaves	50	feature_fraction	0.6
max_bin	175	lambda_l1	0
min_data_in_leaf	11	lambda_l2	0.9

• min_data_in_leaf, the minimum amount of data on a leaf node. It can be used for dealing with overfitting.
• bagging_section, which randomly selects part of the data without resampling, can be used to accelerate training speed and process overfitting.
• bagging_freq, bagging frequency, the set value corresponds to the number of iterations of bagging.
• feature_fraction, if the set value is less than 1, the algorithm will randomly select some features in each iteration.
• lambda_l1, the penalty coefficient of L1 regularization.
• lambda_l2, the penalty coefficient of L2 regularization.
• min_split_gain, the minimum partition gain required by the leaf nodes of the spanning tree.

We perform grid search to get the optimal values of the parameters listed above, as shown in Table 2.

After using LightGBM to generate GBDT, we calculate the global importance of feature j (see formula (6)) and the importance of feature j in a single tree (see formula (7)) through the method proposed by [46],

$$J_j^2 = \frac{1}{M}\sum_{m=1}^{M} J_j^2(T_m), \tag{6}$$

$$J_j^2(T) = \sum_{t=1}^{L-1} i_t^2(v_t = j), \tag{7}$$

where M is the number of trees; L is the number of leaf nodes of the tree; $L-1$ is the number of non-leaf nodes of the tree (the number of constructed trees is a binary tree with left and right children); v_t is the feature associated with node t; i_t^2 is the reduction of square loss caused by node t splitting; and the results are normalized to select the feature whose cumulative importance reaches 99%. See Table 3 for details. After the correlation coefficient, LightGBM and feature importance are processed on the features, 15 features have been dropped.

3) DATA NORMALIZATION

Data normalization is the indexation of statistical data. In a multi-index evaluation system, due to the different nature of the evaluation indexes, they usually have different dimensions and orders of magnitude. When the levels of various indicators are very different, if you directly use the original indicator values for analysis, it will highlight the role of indicators with higher numerical values in the comprehensive analysis, and relatively weaken the role of indicators with lower numerical levels. Therefore, so as to ensure the data's reliability and availability, it is essential to standardize the original index data so that different features have the same scale. In this paper, we use min-max normalization, as shown in formula (8), to make the attribute values fall within the [0, 1], meanwhile, we convert the original data into dimensionless index evaluation values, for the sake of preparing for the comprehensive evaluation and analysis of subsequent modeling.

$$x' = \frac{x - \min}{\max - \min} \tag{8}$$

TABLE 3. The value of the parameter in LightGBM.

Feature	Importance	Normalized importance	Cumulative importance
smeansz	3487.2	0.080071272	0.080071271
sbytes	3426.9	0.078686695	0.158757968
ct_srv_src	2067.5	0.04747286	0.206230827
sload	2041.6	0.046878157	0.253108984
dtcpb	2004.6	0.046028582	0.299137567
stcpb	1887.9	0.043348978	0.342486545
dur	1862.9	0.042774941	0.385261485
tcprtt	1804.1	0.041424806	0.426686291
ackdat	1759.5	0.040400724	0.467087015
synack	1746.2	0.040095336	0.507182351
sinpkt	1582.6	0.036338838	0.543521189
ct_srv_dst	1540.7	0.035376752	0.578897941
ct_dst_src_ltm	1439.0	0.033041569	0.611939510
dbytes	1391.2	0.031944011	0.643883521
djit	1364.7	0.031335532	0.675219053
proto	1273.9	0.029250629	0.704469682
dload	1264.8	0.02904168	0.733511361
dinpkt	1216.1	0.027923456	0.761434817
sjit	1183.6	0.027177208	0.788612024
dmeansz	1108.4	0.025450504	0.814062529
ct_src_ltm	1020.0	0.023420709	0.837483238
ct_dst_ltm	904.9	0.020777843	0.858261081
ct_dst_sport_ltm	677.9	0.015565587	0.873826668
sttl	647.1	0.014858374	0.888685042
ct_state_ttl	605.6	0.013905472	0.902590514
ct_src_dport_ltm	577.2	0.013253366	0.915843880
service	512.3	0.011763166	0.927607046
sloss	470.6	0.010805672	0.938412719
res_body_len	468.9	0.010766638	0.949179357
dloss	447.0	0.010263781	0.959443138
spkts	410.3	0.009421095	0.968864233
dpkts	357.0	0.008197248	0.977061482
state	279.7	0.006422326	0.983483808
dttl	217.8	0.00500101	0.988484818
ct_flw_http_mthd	177.4	0.004073367	0.992558184

Among them, x and x' are separately the original and the normalized data while min and max are the minimum and maximum values of each feature.

4) BINARY PARTICLE SWARM OPTIMIZATION (BPSO)

The essence of feature selection is to select a suitable 0/1 string, which 1 represents the selected feature, while 0 indicates the discarded feature. The length of the string is determined by the number of features in the original dataset. This paper inputs the processed data in section IV-B1, IV-B1 and IV-B3 into BPSO for further feature optimization. Compared with PSO, the BPSO's velocity update formula is shown in (2), and the position update formula adds a sigmoid function on the basis of (3), as shown in (9):

$$x_{ij} = \begin{cases} 1, & rand < sigmoid\left(v_{ij}\right) \\ 0, & otherwise \end{cases} \tag{9}$$
$$sigmoid\left(v_{ij}\right) = \frac{1}{1 + e^{-v_{ij}}}$$

The resulting position is a 0/1 string, and the feature whose result is 1 is the final feature to be selected.

Like LightGBM, BPSO also has some parameters. Based on experience and experimental practice, we have achieved better results by setting the parameters, seeing Table 4.

TABLE 4. Parameter setting of BPSO.

Parameter	Parameter meaning	Value
Classifier	Classifier algorithm selected by BPSO	Random forest
α	A constant in a fitness function that regulates classifier performance and number of features	0.9
c_1	Learning factor, the weight coefficient of the particle tracking its optimal position in history	2
c_2	Learning factor, the weight coefficient of the historical optimal position of the particle tracking group	2
w	Inertia factor	0.7
n_particles	Number of particles	30

The horizontal axis of Figure 11 shows the feature subset of the original dataset, and the vertical axis describes the average accuracy score of the classifier after 10 cross validation between the feature subset and the data class label under the random forest classifier, indicating that although the number of feature subsets is constantly changing, the accuracy rate of the classifier is mostly between 92% and 93.5%, which can reach more than 90%, showing the good performance of the classifier.

Figure 12 is a line chart between the number of iterations and the cost optimized by BPSO after 10 iterations, from which we observe that the BPSO has reached the convergence state by the sixth iteration.

The line chart of the cost of finding the optimal location globally and that of the mean value of particle searching for the optimal location with the number of iterations is shown in Figure 13.

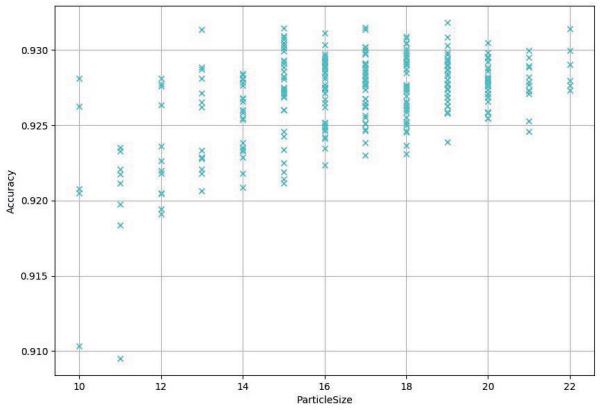


FIGURE 11. Scatter plot of feature subset's number and model accuracy.

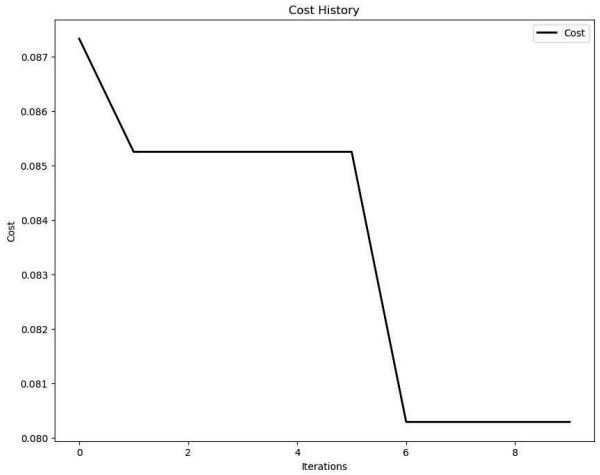


FIGURE 12. Cost of BPSO optimization with different iterations.

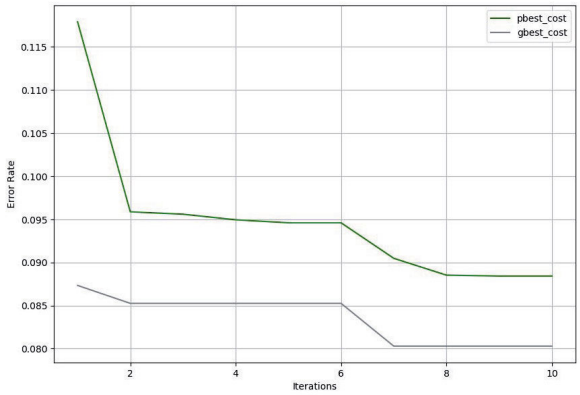


FIGURE 13. Cost of BPSO optimization with different iterations.

After the above four steps of feature filtering, Table 5 shows the final extracted 23 features (excluding class label feature).

TABLE 5. Feature extraction results.

Feature	Data type	Feature	Data type
dur	float	stcpb	integer
proto	nominal	dtcpb	integer
service	nominal	tcprtt	float
state	nominal	ackdat	float
spkts	integer	smeansz	integer
dpkts	integer	dmeansz	integer
sload	float	res_body_len	integer
dload	float	ct_state_ttl	integer
sinpkt	float	ct_dst_ltm	integer
dinpkt	float	ct_dst_sport_ltm	integer
sjit	float	ct_src_ltm	integer
djit	float		

C. MODEL TRAINING AND DATA DETECTION

Model building is the core part of the entire intrusion detection. It trains the preprocessed normalized data to produce a robust model, which is conducive to the judgment of test data behavior.

Radial basis function (RBF) is a scalar function that is symmetric along the radial direction. It can realize nonlinear mapping and the parameters have little effect on the model complexity. Therefore, this paper uses RBF as the kernel function of OCSVM. We import the preprocessed data into OCSVM for model training and data behavior detection. The experimental results are detailed in Section V-B.

V. EXPERIMENTAL EVALUATION

A. EVALUATION METRICS

In the field of intrusion detection, the prediction of data includes four cases: TP, FN, TN, FP, and their respective definitions are as follows:

1) *TP (True Positive):* Both the actual label and the predicted label are positive;
2) *FN (False Negative):* The actual label is positive, while the predicted label is negative;
3) *TN (True Negative):* Both the actual label and the predicted label are negative;
4) *FP (False Positive):* The actual label is negative, while the predicted label is positive;

TABLE 6. Confusion matrix.

Predicted / Actual	Positive	Negative	Total
Positive	TP	FN	P(Actual)
Negative	FP	TN	N(Actual)
Total	P(Predicted)	N(Predicted)	P+N

The confusion matrix composed of the above four situations is shown in Table 6.

According to the above definition, there are usually three indicators of detection rate (DR), accuracy and false alarm rate (FAR) to evaluate IDS, as follows:

1) Detection rate (DR), the correct proportion of all positive instances, is a measure of coverage, which measures the recognition ability of the classifier for positive instances, as shown in (10).

$$DR = \frac{TP}{TP + FN} \tag{10}$$

2) Accuracy, the proportion of the correct result of the prediction to the total number of samples, which can be used to judge the overall correct rate of the sample classification by the model, is described as (11).

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{11}$$

3) False alarm rate (FAR), which is predicted positive but actually negative, accounting for the proportion of all actual negative, is defined as (12).

$$FAR = \frac{FP}{TN + FP} \tag{12}$$

B. EXPERIMENT SETUP

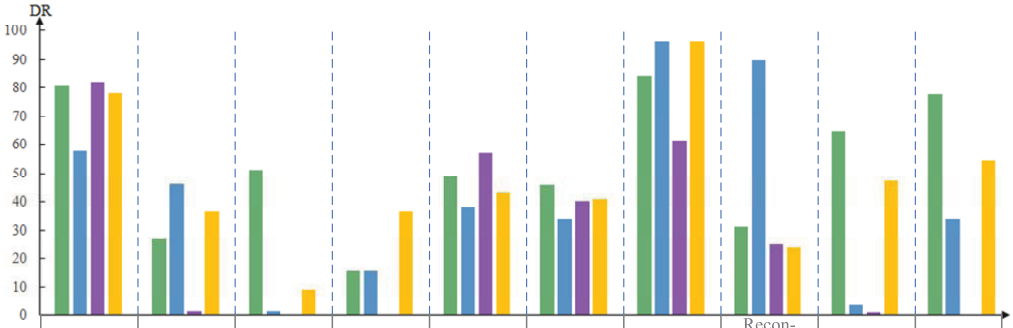
PSO-LightGBM intrusion detection is tested with 23 features in Table 4. The technology is implemented on a PC with 8GB RAM and i5 processor, coded by Python in feature extraction and C ++ in model training and detection phase. In order to speed up the model training, we conducted a 20% stratified sampling of the training set and test set of UNSW-NB15 according to the "attack_cat" label. The data distribution after sampling is shown in the Table 7 below.

TABLE 7. Confusion matrix.

Class label	Class	20%Training set		20%Test set	
		Count	Frequency	Count	Frequency
0	Normal	11200	0.3194	7400	0.4494
1	Analysis	400	0.0114	135	0.0082
2	Backdoor	349	0.0100	117	0.0071
3	DoS	2453	0.0699	818	0.0497
4	Exploits	6679	0.1905	2226	0.1352
5	Fuzzers	3637	0.1037	1212	0.0736
6	Generic	8000	0.2281	3774	0.2292
7	Reconn-aissance	2098	0.0598	699	0.0425
8	Shellcode	227	0.0065	76	0.0046
9	Worms	26	0.0007	9	0.0005
Total		35069	1	16466	1

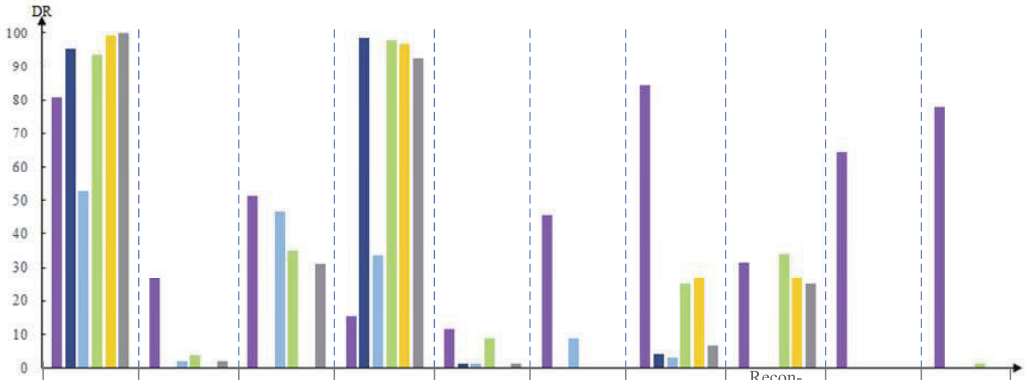
C. RESULTS AND ANALYSIS

Figure 14 shows the detection rate of the proposed intrusion detection model on Normal and 9 types of attack data, compared with CMLW [47], TSDL [48] and Threshold-optimized CNNs [49]. Among them, PSO-LightGBM has an excellent detection rate for most of the data's type, especially in the



	Normal	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Recon-naissance	Shellcode	Worms
PSO-LightGBM	80.85	26.67	51.28	15.40	48.56	45.63	84.34	31.33	64.47	77.78
CMLW	57.68	46.09	1.54	15.60	37.68	33.67	96.20	89.76	3.70	33.67
TSDL	82.00	1.34	0	0.44	57.14	40.30	61.21	24.89	0.85	0
Threshold-optimized CNNs	78.08	36.19	8.92	36.22	43.17	40.88	96.19	23.80	47.35	54.55

FIGURE 14. Comparison results of PSO-LightGBM with state-of-the-art IDS based on UNSW-15 dataset in terms of DR(%).



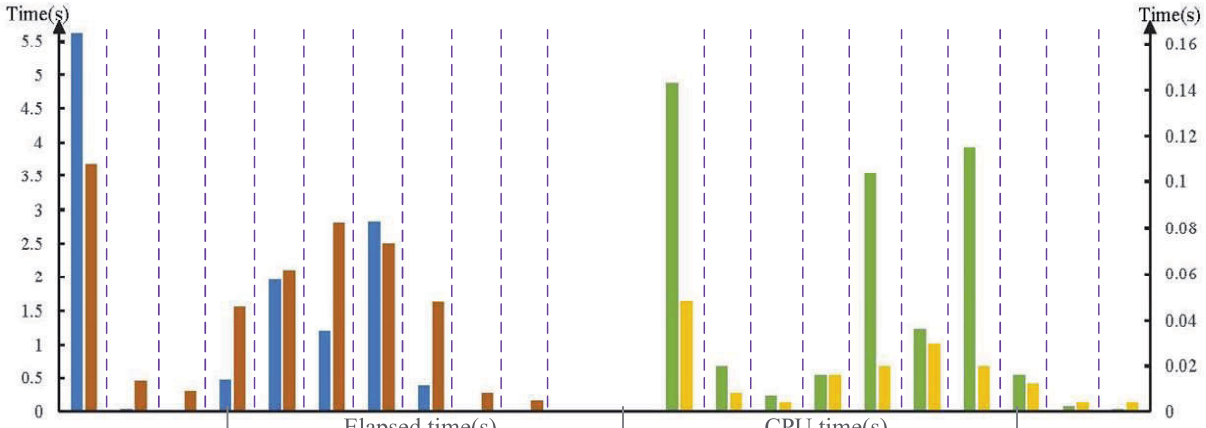
	Normal	Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Recon-naissance	Shellcode	Worms
PSO-LightGBM	80.85	26.67	51.28	15.40	11.64	45.63	84.34	31.33	64.47	77.78
LR	95.40	0	0	98.40	1.10	0	3.80	0	0	0
NB	52.70	1.80	46.50	33.50	1.30	8.60	2.80	0	0	0
KNN	93.60	3.50	34.90	97.70	8.80	0	24.80	33.80	0	1.15
AB	99.20	0	0	96.70	0	0	26.90	27.10	0	0
SVM	99.80	1.70	30.90	92.60	1.28	0	6.80	25.00	0	0

FIGURE 15. Comparison results of PSO-LightGBM with standard machine learning algorithm based on UNSW-15 dataset in terms of DR(%).

detection of Backdoor, Shellcode and Worms, the three low frequency attacks, reached 51.28%, 64.47% and 77.78%, respectively, which is much higher than the other three IDSs listed, showing the excellent anomaly detection ability of this model. DoS usually uses the means of temporarily interrupting or suspending the connection to the Internet host service to make the server or network resources unavailable to the user, and further destroys the computer resources through the memory, preventing the authorization request from accessing the device, which directly leads to the low detection rate of the abnormal data [6], [19]. Reconnaissance can be regarded as a kind of probe. Its main task is to collect relevant computer network information to evade its security control attacks. A series of false alarms will inevitably be generated during the detection process, so its detection rate is not high [19].

In the process of comparing the model with standard machine learning algorithms, for instance, logistic regression (LR), naive bayes (NB), k-nearest neighbor (KNN), adaboost (AB), and support vector machine (SVM), not only can PSO-LightGBM recognize low frequency data, but also has more advantages than the traditional machine learning method in the detection of high frequency data such as Normal and Generic, as shown in Figure 15.

For IDS, only considering its DR is too one-sided, so this paper employs accuracy and FAR to further evaluate the model. Table 8 shows the comparison between



	Elapsed time(s)		CPU time(s)	
	Training phase	Testing phase	Training phase	Testing phase
Normal	5.612	3.682	0.143	0.048
Analysis	0.029	0.459	0.020	0.008
Backdoor	0.015	0.299	0.007	0.004
DoS	0.473	1.557	0.016	0.016
Exploits	1.957	2.100	0.104	0.02
Fuzzers	1.195	2.187	0.036	0.03
Generic	2.824	2.502	0.115	0.02
Recon-naissance	0.383	1.620	0.016	0.012
Shellcode	0.010	0.271	0.002	0.004
Worms	0.004	0.163	0.001	0.004

FIGURE 16. Time cost of various data under UNSW-NB15 dataset.

PSO-LightGBM and other IDS in terms of accuracy and FAR, seeing that the our model performs well in the evaluation of these two indicators, especially the significant reduction of FAR, which exhibits the good robustness and robustness. As shown in Table 9, PSO-LightGBM also shows strong detection ability in comparion with single machine learning algorithm, manifesting the importance of deploying IDS with mixed technologies in IoT.

In addition to considering the performance of IDS in DR, FPR and accuracy, we also analyze the elapsed time and CPU time cost by the proposed IDS in the model training and testing phases. From Figure16, we can see that the time cost by different types of data in the training and testing stages is proportional to the number of such data. According to the statistics of data in Table 7, Normal data accounts for the largest proportion in the training set and the test set, so it costs more time, whereas the low frequency data, Analysis, Backdoor, Shellcode and Worms, need less time in the training stage and the test stage. We run the model

TABLE 8. Comparison results of PSO-LightGBM with state-of-the-art IDS based on UNSW-15 dataset in accuracy and FAR.

Method	Accuracy(%)	FAR(%)
PSO-LightGBM	86.68	10.62
Expectation-maximization [19]	78.47	23.79
TSDL_DT [48]	85.56	15.78
TSDL_ANN [48]	81.34	21.13
TSDL_LR [48]	83.15	18.48
Two-stage classifier [50]	85.78	15.64
I-ELM [41]	67.01	31.96
I-ELM+A-PCA [41]	70.51	35.09
IGAN [51]	82.52	/

10 times, and the time required after averaging is the elapsed time and CPU time required for the entire training process and testing process, as shown in Table 10. Because there is only one kind of data in the training process of OCSVM,

TABLE 9. Comparison results of PSO-LightGBM with standard machine learning algorithm based on UNSW-15 dataset in accuracy and FAR.

Method	Accuracy(%)	FAR(%)
PSO-LightGBM	86.68	10.62
Decision Tree [19]	85.56	15.78
Logistics Regression [19]	83.15	18.48
Naïve Bayes [19]	82.07	18.56
Multilayer Perceptron [48]	81.30	21.15
Artificial Neural Network [19]	81.34	21.36
EM clustering [19]	78.47	23.79

TABLE 10. Total time cost under UNSW-NB15 dataset.

Dataset	Total Training phase time(s)		Total Testing phase time(s)	
	Elapsed time	CPU time	Elapsed time	CPU time
UNSW-NB15	12.502	0.46	3.682	0.048

we take the total training time of all kinds of data as the time cost of the whole IDS in the training phase. Therefore, the proposed model uses 12.502s to process 35069 records in the training phase, that is, 2805 records per second. In the test phase, the set containing different types of data is input into the model for simultaneous testing. Because of the uneven distribution of various types of data, the Normal data, which accounts for the largest proportion of data, is finally detected. Therefore, the test time of normal data is 3.682s, which means that IDS can detect 4472 records per second.

VI. CONCLUSION AND FUTURE WORK

In this article, an intrusion detection model based on particle swarm optimization for IoT is proposed. In the data preprocessing stage, the IDS performs numerical type conversion, correlation coefficient calculation and normalization on the characteristics of the dataset, and further introduces LightGBM on the basis of particle swarm optimization, through adjustment of parameters, so that the number of features is reduced from 42 to 23; in the model training stage, the extracted feature data is input into OCSVM to complete the recognition of normal and abnormal data. In this study, a 20% random sampling is performed on the original training set and test set of UNSW-NB15, and the sampled data is used as the experimental benchmark dataset. In the case of detection rate, most of the existing IDS and traditional machine learning algorithms maintain high detection rates for high frequency data, such as Normal and Generic, but it is almost impossible to detect Backdoor, Shellcode, and Worms. PSO-LightGBM not only does not reduce the detection ability of high frequency data, but greatly increases the detection rate of Backdoor, Shellcode and Worms to reach 51.28%, 64.47% and 77.78%, respectively. In terms of accuracy and FAR, PSO-LightGBM performs equally well and it does not consume much time in the process of data detection and identification. Through simulation experiments, it is shown that our model has good robustness and generalization in the detection of a variety of abnormal data, laying the foundation for practical application in IoT.

In future work, we plan to integrate the existing model with multi-task [52] learning in the model training stage to better improve the performance indicators of the model.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[2] M. El-hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, "A survey of Internet of Things (IoT) authentication schemes," *Sensors*, vol. 19, no. 5, p. 1141, Mar. 2019.

[3] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use AI to enhance security?" *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 41–49, Sep. 2018.

[4] J. M. Ehrenfeld, "WannaCry, cybersecurity and health information technology: A time to act," *J. Med. Syst.*, vol. 41, no. 7, p. 104, Jul. 2017.

[5] D. M. Hawkins, *Identification of Outliers*, vol. 11. Springer, 1980.

[6] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[7] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Proc. IEEE Symp. Secur. Privacy*, Sep. 1999, pp. 120–132.

[8] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proc. SIAM Int. Conf. Data Mining*, May 2003, pp. 25–36.

[9] A. S. A. Aziz, A. T. Azar, A. E. Hassanien, and S. E.-O. Hanafy, "Continuous features discretization for anomaly intrusion detectors generation," in *Soft Computing in Industrial Applications*. Springer, 2014, pp. 209–221.

[10] S. Pontarelli, G. Bianchi, and S. Teofili, "Traffic-aware design of a high-speed FPGA network intrusion detection system," *IEEE Trans. Comput.*, vol. 62, no. 11, pp. 2322–2334, Nov. 2013.

[11] A. Abraham, C. Grosan, and C. Martin-Vide, "Evolutionary design of intrusion detection programs," *IJ Netw. Secur.*, vol. 4, no. 3, pp. 328–339, 2007.

[12] C.-F. Tsai and C.-Y. Lin, "A triangle area based nearest neighbors approach to intrusion detection," *Pattern Recognit.*, vol. 43, no. 1, pp. 222–229, Jan. 2010.

[13] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.

[14] N. Moustafa and J. Slay, "A hybrid feature selection for network intrusion detection systems: Central points," 2017, *arXiv:1707.05505*. [Online]. Available: http://arxiv.org/abs/1707.05505

[15] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: Statistical decision-making using finite Dirichlet mixture models," in *Data Analytics and Decision Support for Cybersecurity*. Springer, 2017, pp. 127–156.

[16] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," in *Proc. Int. Conf. Signal Process. Commun. Eng. Syst.*, Jan. 2015, pp. 92–96.

[17] H. Hindy, D. Brosset, E. Bayne, A. K. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A taxonomy of network threats and the effect of current datasets on intrusion detection systems," *IEEE Access*, vol. 8, pp. 104650–104675, 2020.

[18] R. Xu, R. An, and X. Geng, "Research intrusion detection based PSO-RBF classifier," in *Proc. IEEE 2nd Int. Conf. Softw. Eng. Service Sci.*, Jul. 2011, pp. 104–107.

[19] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J., A Global Perspective*, vol. 25, nos. 1–3, pp. 18–31, Apr. 2016.

[20] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4. Nov./Dec. 1995, pp. 1942–1948.

[21] L. Ning and Z. Jianhua, "Intrusion detection research based on improved PSO and SVM," in *Proc. Int. Conf. Autom. Control Artif. Intell. (ACAI)*, 2012, pp. 1263–1266.

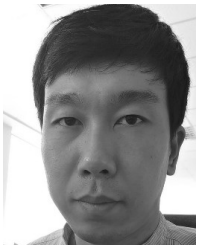
[22] X. Tan, S. Su, Z. Zuo, X. Guo, and X. Sun, "Intrusion detection of UAVs based on the deep belief network optimized by PSO," *Sensors*, vol. 19, no. 24, p. 5529, Dec. 2019.

[23] S. M. Vieira, L. F. Mendonça, G. J. Farinha, and J. M. C. Sousa, "Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients," *Appl. Soft Comput.*, vol. 13, no. 8, pp. 3494–3504, Aug. 2013.
[24] S. Srinoy, "Intrusion detection model based on particle swarm optimization and support vector machine," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Apr. 2007, pp. 186–192.
[25] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 3146–3154.
[26] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.
[27] S. Tyree, K. Q. Weinberger, K. Agrawal, and J. Paykin, "Parallel boosted regression trees for Web search ranking," in *Proc. 20th Int. Conf. World Wide Web WWW*, 2011, pp. 387–396.
[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, and J. Vanderplas, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[29] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.
[30] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, Jan. 2004.
[31] F. Hosseinpour, P. V. Amoli, J. Plosila, T. Hämäläinen, and H. Tenhunen, "An intrusion detection system for fog computing and IoT based logistic systems using a smart data approach," *Int. J. Digit. Content Technol. Appl.*, vol. 10, 2016.
[32] M. Nobakht, V. Sivaraman, and R. Boreli, "A host-based intrusion detection and mitigation framework for smart home IoT using OpenFlow," in *Proc. 11th Int. Conf. Availability, Rel. Secur. (ARES)*, Aug. 2016, pp. 147–156.
[33] B. Alotaibi and K. Elleithy, "A majority voting technique for wireless intrusion detection systems," in *Proc. IEEE Long Island Syst., Appl. Technol. Conf. (LISAT)*, Apr. 2016, pp. 1–6.
[34] H. H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha, and K.-K.-R. Choo, "A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 2, pp. 314–323, Apr. 2019.
[35] N. Moustafa, B. Turnbull, and K.-K.-R. Choo, "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4815–4830, Jun. 2019.
[36] J. Roux, E. Alata, G. Auriol, V. Nicomette, and M. Kaaniche, "Toward an intrusion detection approach for IoT based on radio communications profiling," in *Proc. 13th Eur. Dependable Comput. Conf. (EDCC)*, Sep. 2017, pp. 147–150.
[37] Q. Tian, D. Han, K.-C. Li, X. Liu, L. Duan, and A. Castiglione, "An intrusion detection approach based on improved deep belief network," *Int. J. Speech Technol.*, vol. 50, no. 10, pp. 3162–3178, Oct. 2020.
[38] I. Benmessahel, K. Xie, and M. Chellal, "A new evolutionary neural networks based on intrusion detection systems using multiverse optimization," *Int. J. Speech Technol.*, vol. 48, no. 8, pp. 2315–2327, Aug. 2018.
[39] S. M. Hosseini Bamakan, H. Wang, and Y. Shi, "Ramp loss K-Support vector classification-regression; a robust and sparse multi-class approach to the intrusion detection problem," *Knowl.-Based Syst.*, vol. 126, pp. 113–126, Jun. 2017.
[40] M. AL-Hawawreh, N. Moustafa, and E. Sitnikova, "Identification of malicious activities in industrial Internet of Things based on deep learning models," *J. Inf. Secur. Appl.*, vol. 41, pp. 1–11, Aug. 2018.
[41] J. Gao, S. Chai, B. Zhang, and Y. Xia, "Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis," *Energies*, vol. 12, no. 7, p. 1223, Mar. 2019.
[42] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Sep. 2016, pp. 139–144.
[43] H. Zhang, L. Huang, C. Q. Wu, and Z. Li, "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset," *Comput. Netw.*, vol. 177, Aug. 2020, Art. no. 107315.
[44] H. Zhang, C. Q. Wu, S. Gao, Z. Wang, Y. Xu, and Y. Liu, "An effective deep learning based scheme for network intrusion detection," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 682–687.
[45] J. Benesty, J. Chen, and Y. Huang, "On the importance of the pearson correlation coefficient in noise reduction," *IEEE Trans. Audio, Speech, Language Process.*, vol. 16, no. 4, pp. 757–765, May 2008.
[46] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
[47] K. Wu, Z. Chen, and W. Li, "A novel intrusion detection model for a massive network using convolutional neural networks," *IEEE Access*, vol. 6, pp. 50850–50859, 2018.
[48] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "A novel two-stage deep learning model for efficient network intrusion detection," *IEEE Access*, vol. 7, pp. 30373–30385, 2019.
[49] J. Luo, S. Chai, B. Zhang, Y. Xia, J. Gao, and G. Zeng, "A novel intrusion detection method based on threshold modification using receiver operating characteristic curve," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 14, Jul. 2020, Art. no. e5690.
[50] W. Zong, Y.-W. Chow, and W. Susilo, "A two-stage classifier approach for network intrusion detection," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, Springer, 2018, pp. 329–340.
[51] S. Huang and K. Lei, "IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks," *Ad Hoc Netw.*, vol. 105, Aug. 2020, Art. no. 102177.
[52] F. Xiong, B. Sun, X. Yang, H. Qiao, K. Huang, A. Hussain, and Z. Liu, "Guided policy search for sequential multitask learning," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 1, pp. 216–226, Jan. 2019.

JINGYU LIU received the B.S. degree from Shandong Normal University, China, in 2018. She is currently pursuing the M.S. degree with the Shenyang Institute of Computing Technology, University of Chinese Academy of Sciences. Her research interests include intrusion detection, the Internet of Things, and machine learning.

DONGSHENG YANG received the B.S. degree from Jilin University, China, in 1986. He is currently a Professor and a Ph.D. Advisor with the Shenyang Institute of Computing Technology, Chinese Academy of Sciences. His research interests include the Internet of Things, machine learning and pattern recognition, real-time control systems, and numerical control. His awards and honors include the Technology Progress of Chinese Academy of Science and the Liaoning Province Technology Progress.

MENGJIA LIAN received the B.S. degree from Shenyang Normal University, China, in 2014. She is currently pursuing the Ph.D. degree with the Shenyang Institute of Computing Technology, University of Chinese Academy of Sciences. Her research interests include the Internet of Things, and middleware and numerical control.



MINGSHI LI was born in 1990. He received the M.S. degree from the University of Chinese Academy of Sciences, in 2017. He is currently pursuing the Ph.D. degree with the Shenyang Institute of Computing Technology, University of Chinese Academy of Sciences. His research interests include the Internet of Things and communication security.