Hindawi
Mobile Information Systems
Volume 2022, Article ID 3850582, 10 pages
https://doi.org/10.1155/2022/3850582

Research Article

Network Intrusion Detection Method Based on Improved CNN in Internet of Things Environment

Yulin Wang, Jinheng Wang, and Honglin Jin

School of Computer Science and Engineering, Guangzhou Institute of Science and Technology, Guangzhou, Guangdong 510540, China

Correspondence should be addressed to Jinheng Wang; wyl@gzist.edu.cn

Received 25 March 2022; Revised 25 April 2022; Accepted 20 May 2022; Published 8 June 2022

Academic Editor: Imran Shafique Ansari

In view of most existing intrusion detection technologies that cannot meet the actual needs of the Internet of Things and facing the problems of poor detection effect of complex network intrusion methods, a network intrusion detection method based on deep learning algorithm in the environment of the Internet of Things is proposed. Firstly, the Internet of Things intrusion detection model is constructed based on edge computing, in which the concept of gated convolution is introduced to improve the convolution neural network model. Data passes through convolution layer, pooling, dropout, full connection, and Softmax function to realize multiclassification. Finally, the Focal Loss function is used to modulate the training ratio of positive and negative samples to solve the problem of uneven distribution of sample data. The proposed algorithm is demonstrated experimentally based on KDD99 data set. The results show that the accuracy, precision, recall, and $F1$ values are 92.14%, 95.97%, 90.89%, and 90.03%, which are better than other comparison algorithms. The proposed method can better meet the needs of Internet of Things intrusion detection.

1. Introduction

Internet of Things (IoT) is one of the hottest technologies in the modern science and technology. Since 2005, IoT has been widely used in various fields, such as industry, automobile, medical treatment, energy, and health care. Other industries have also begun to integrate with IoT; for example, sensors and robots are connected with multiple devices [1, 2]. On the one hand, the wide application of IoT improves work efficiency and brings convenience to people's life; on the other hand, the network is facing increasingly serious security threats. The traditional IoT system is physically isolated from the outside world. It mainly focuses on the stability and functional security of the system but lacks consideration of information security. Therefore, it is very important to design and implement the defense mechanism of IoT [3].

According to the vulnerability types and attack methods of the IoT, defense mechanisms and detection systems at different levels and aspects have been produced [4]. According to the hierarchy of defense mechanism, the first defense line is the firewall of the IoT, which avoids illegal attacks on the external network by deploying security gateways and routers, and the second defense line is intrusion detection system (IDS), which detects unreliable or unauthorized behaviors in the network [5, 6]. From the perspective of detection methods, IDS can be divided into misuse detection and anomaly detection. However, with the diversified application of IoT terminal equipment and the complex expansion of the network environment, it is more difficult to explore the potential security vulnerabilities in the network, and the methods of network attack are more difficult to predict. Most of the existing intrusion detection technologies cannot meet the actual needs. It exposes the problem that IDS has poor ability of adapting to the current network environment [7, 8].

As a representative algorithm of deep learning, Convolutional Neural Network (CNN) can automatically learn data features, relationship mapping, and other pieces of information in the process of network transmission from

data samples. CNN has achieved some achievements in the fields of image processing, natural language processing, and speech recognition [9]. Therefore, in order to improve IoT's ability of dealing with various network attacks, CNN is introduced to design an IoT intrusion detection algorithm based on deep learning in edge computing environment.

Because cloud computing requires high bandwidth and has large transmission delay, the proposed algorithm uses the big data processing idea of edge computing to process the more data on the computing resources close to the data source, so as to reduce the pressure of network transmission bandwidth and improve the overall efficiency of data processing. The experimental results based on KDD99 data set show that the time-consumption of receiving and storing data is 236 ms and 157 ms, respectively, which greatly speeds up the efficiency of data processing.

The remaining chapters of this paper are arranged as follows: the second chapter introduces the relevant research in the field of IoT intrusion detection. The third chapter introduces the construction of the data set and the process of data preprocessing. The fourth chapter introduces the proposed sensor data intrusion detection algorithm. In Chapter 5, experiments are designed to verify the performance of the proposed algorithm. The sixth chapter is the conclusion.

2. Related Work

As an active defense technology in network system, intrusion detection has gradually attracted the attention of researchers [10]. Authors of [11] used singular value decomposition technology to reduce the dimension of features, further reconstruct features to form reduced features, and select feature vectors. The reconstruction loss is used to determine the intrusion category of a given network feature, and a high detection accuracy is obtained. However, there are still great difficulties for the existing complex intrusion detection. In [12], data mining was applied to classify abnormal data and analyzed the abnormal traffic and data by using all the features that are actually irrelevant to the classification target to improve the accuracy of intrusion detection. However, the amount of calculation of data mining is large, and the cost of intrusion detection is high.

With the spread of new viruses and the emergence of intrusion behavior, intrusion detection system also continues to be innovated. With the rapid development of intelligent algorithms, data mining and machine learning technology are applied to the field of intrusion detection, and innovative ideas of pattern recognition related algorithms are introduced into intrusion detection system, which also enhances the detection effect [13]. In [14], an efficient clustering technology of adaptive chicken colony optimization algorithm is proposed for cluster head selection. Combined with the two-stage classification technology of adaptive support vector machine (SVM) classification, malicious sensor nodes are reported, which minimizes time consumption and improves network lifetime and scalability. Abraham A et al. (2019) proposed an intrusion detection model based on the framework of double sparse convolution matrix [15]. The model uses the close correlation of nonnegative matrix decomposition to identify the hidden pattern of events and obtains a high detection accuracy. However, in dealing with the above technical challenges, the intrusion detection technology based on machine learning involves large number calculations of mathematical formula. The traditional machine learning methods still have great limitations [16].

In recent years, intrusion detection methods based on artificial intelligence such as deep learning are widely used in defensing different network attacks. The excellent autonomous feature learning ability of deep learning has been recognized. Therefore, deep learning has certain advantages in intrusion detection [16]. In addition, due to massive high dimension and nonlinear data in the communication process, diversity, and complexity of network attacks and host viruses, intrusion detection faces new problems [17]. The deep learning method pays more attention to the extraction of representation features between amounts of data and has stronger feature extraction ability when dealing with data with high dimension, high complexity, and noise. Therefore, the deep learning method has been widely used in intrusion detection [18]. Azawii A et al. (2019) proposed an anomaly intrusion detection data classification scheme based on deep learning, deeply analyzed the discriminant, hybrid, and generative intrusion detection schemes, and deeply studied the existing data sets and frameworks [19]. However, the scheme needs more training times and cost to be realized. Riyaz B et al. (2020) proposed a new intrusion detection system, which provides the security of data communication by effectively identifying and detecting intruders in wireless networks [20]. A new conditional random field and a feature selection algorithm based on linear correlation coefficient are used to select the features with the greatest contribution, and the existing CNN is used to classify the features. However, due to the high dimension of network data, it is difficult to realize multiscale extraction of local features. Aiming at the problems of low accuracy of intrusion detection, the authors of [21] proposed a traffic anomaly detection model, which combines the two-way long-term and short-term memory network and attention mechanism. In addition, multiple convolution layers are used to capture the local features of traffic data, but there are some limitations in the processing of traffic data from different malware. In view of the above shortcomings, the IoT intrusion detection based on deep learning in the edge computing environment is proposed. The proposed method uses improved CNN model to complete data multiclassification at the edge of the network and to realize IoT intrusion detection.

3. Data Set Construction and Data Preprocessing

3.1. Data Set Construction. The data set used is an authoritative public data set in the field of intrusion detection, KDD99 data set, which simulates the network traffic generated by attackers in different complex networks. There are about 5 million pieces of data in the whole data set, and each piece of data represents a network behavior. The network

behavior may be an attack behavior or a normal behavior, which is determined by the data label [22, 23]. In this data set, network behaviors are divided into normal behaviors (Normal) and attack behaviors (Attack).

Each record in the KDD99 data set identifies a network connection. Each network connection is represented by 41 features and stored in the CSV file. The last column of each record is a label, representing whether the network record is normal or abnormal. These 41 features are roughly divided into four aspects: first, some basic attributes of network connection, with 9 features, such as protocol type and number of bytes; second, some content attributes of network connection have 13 features in total, which are mainly used to detect the attacks that do not occur frequently and are highly hidden, such as U2R and R2L. These two types of attacks will be hidden in the data load of data packets. This type of data packets is not different from normal data packets from the outside. Therefore, researchers extract features that can reflect intrusion behavior, such as the number of login failures, to characterize such attacks; third, there is the behavior statistical attribute of network connection, which describes the statistics of a behavior in a continuous period; fourth, there is the traffic statistical attribute, which describes the correlation between some behaviors in the past.

Each data in KDD99 data set has 41 features, including 38 numerical features and 3 symbolic features. Five records were extracted from the data set, representing four different types of attack behavior and one normal behavior. The specific data information is as follows: each record is composed of 41 dimensional features and a label separated by commas. The 2nd, 3rd, and 4th dimensions are symbolic features, the last dimension is label, and the rest are numerical features.

3.2. Data Preprocessing. From the application level, network-based intrusion detection technology is to identify and respond to potential abnormal behaviors in the network. From the technical level, network-based intrusion detection technology is essentially a process of feature extraction and classification of network traffic data. There are many factors affecting the classification performance of network-based IDS, including the source of network traffic data, the probability distribution of sample category, the preprocessing of data samples, the construction, and training of classification model [24]. Therefore, before constructing an intrusion detection model with high performance, it is necessary to effectively preprocess the sample data. Data preprocessing can better reflect the internal relationship between the data itself and the sample data.

For the intrusion detection KDD99 data set, the data types of inherent attributes and category contain integer type, string type, floating point type, and Boolean type, and the data probability distribution of each attack type is uneven. Therefore, from the overall viewpoint, data preprocessing is divided into five steps: data set cleaning, sample attribute type conversion, sample category equalization, sample attribute normalization, and sample gray image conversion.

3.2.1. Data Set Cleaning. There are two purposes of data cleaning: the first is to filter the redundant data in the data set and clear the data with multiple duplicate records; the second is to fill in the attributes that may have null values in the data set to prevent affecting the subsequent data preprocessing process.

3.2.2. Sample Attribute Type Conversion. The main goal is to convert the inherent attributes and category of string data types in the data set into integer or higher dimensional binary data types.

The KDD99 data set has 32 integer attributes, 6 Boolean attributes, 3 string attributes, and 1 string category. The three string attributes are protocol attribute, service attribute, and flag attribute. Taking the protocol attribute as an example, its values can be ICMP, TCP, and UDP. After data type conversion, the corresponding values are 0, 1, and 2, respectively. The value range and classification of the attack type are shown in Table 1. The string type attribute in the category is transformed into an integer attribute. The normal category is transformed into 0, and the other 22 categories are transformed into 1 to 22 by ascending order of strings.

3.2.3. Sample Category Equalization. The samples of various categories in the data set are prone to imbalance in global statistics, and the proportion of minority samples in the whole data set is very small. It makes minority samples learn as wrong samples in the process of model training, and it is difficult to correct the misclassification of minority samples in the future. The common methods of sample category equalization are divided into two categories. The first is oversampling, which is used to reduce a class of samples with large probability distribution; the second is undersampling, which is used to increase a class of samples with small probability distribution. The sample equalization method used in the proposed algorithm is SMOTE algorithm. The algorithm is to synthesize new minority samples by using the nearest neighbor samples of minority samples.

3.2.4. Sample Attribute Normalization. The purpose of normalizing the sample data is to map the data with too large or too small order of magnitude in the data set to the same order of magnitude, which facilitates the cross operation between sample attributes and the correlation between different attributes. After the sample attribute type conversion, all attributes of each data record in the data set are numerical. For the continuous numerical attributes, due to the large difference in the value range between attributes, it greatly affects the decision-making of the neural network model in the training process. Therefore, it is necessary to normalize all continuous numerical attributes. The normalization method of sample attributes is as follows: there are a set of data $x = \{x_1, x_2, \ldots, x_n\}$, in which the maximum and minimum values are $x_{\max}$ and $x_{\min}$, respectively. Suppose that x_i is the normalized value. The calculation is as follows:

Table 1: Experimental results of different values of α and τ.

τ	α	Accuracy rate/%	Detection rate/%	False positive rate/%
0	0.6	97.39	94.72	0.72
1	0.4	98.65	96.64	0.68
2	0.2	98.41	97.28	0.31
5	0.2	97.53	96.13	1.09

$$x_i = \frac{x_i - x_{\max}}{x_{\max} - x_{\min}}, \quad x \in [x_{\min}, x_{\max}], i \in [1, n]. \tag{1}$$

The value range of x_i calculated by (1) is mapped to numbers between 0 and 1.

3.2.5. Sample Gray Image Conversion. After the above data preprocessing steps, each attribute in each sample data is numerical type, and the values are mapped to the same interval. The input data of CNN model is usually a two-dimensional matrix, so each sample data needs to be transformed into a two-dimensional matrix, that is, a sample gray image. Each attribute represents a pixel of gray image. The larger the attribute value is, the closer the pixel is to black.

Each sample data read by computer is expressed as a $1 \times n$ one-dimensional vector, where n is the number of inherent attributes. Set $m = \sqrt{n}$ to represent the size of gray image. Because $n \geq m^2$, some attributes cannot be converted into pixels; it is necessary to reduce the dimension of one-dimensional vector. The dimensionality reduction operation uses the variance coefficient as the screening basis, and the function is defined as

$$V' = \frac{\sigma}{\mu}, \tag{2}$$

where σ is the standard deviation and μ is the mean value.

Therefore, after comparison, the dimension with small variance coefficient is taken out and the attributes in the feature set are retained. The $m \times m$ gray image transformed from the $1 \times n$ sample data can be used as the input of CNN model, and the features can be extracted effectively by using the advantage of CNN spatial invariance.

4. Sensor Data Intrusion Detection Algorithm

4.1. Data Architecture Based on Edge Computing. The data extraction model based on heterogeneous devices extracts various forms of heterogeneous data from the IoT data sources of multiple enterprises and saves these multisource heterogeneous data to the integrated Big Data Platform (BDP). BDP supports the whole process of intelligent big data analysis from data extraction, data processing, data mining, and visualization [25, 26].

Because the linear growth of centralized cloud computing capacity has been unable to meet the processing requirements of fast-growing edge data, it will become more and more infeasible technically and economically to concentrate the continuous-growing edge data to one or several data computing centers to complete the corresponding data computing tasks. To this end, edge computing is introduced, some data computing tasks of BDP are migrated accordingly, and an edge layer data processing node is established near the data source to complete the data processing tasks nearby. The overall system architecture is shown in Figure 1.

Taking the specific scenario of the research problem as an example, it is considered to establish the edge layer data node near the sensing data acquisition frond end to complete the anomaly detection task of relevant data while receiving the sensing data.

4.2. Intrusion Detection Algorithm of Internet of Things Based on Deep Learning. Intrusion detection is essentially a classification problem, so deep learning algorithm can be applied to it. Firstly, the data are trained by supervised learning method to get a better classification model, and then the trained classification model is used to predict the unknown data.

In CNN model, the input data in input layer is usually two-dimensional data, while the intrusion detection data is one-dimensional data. Therefore, in terms of convolution selection, one-dimensional convolution method is adopted to convolute the intrusion detection data. At present, CNN is mainly used in the fields of graphics, images, and natural language processing. There is no good CNN model in the field of intrusion detection. Therefore, a CNN network architecture for intrusion detection is designed, as shown in Figure 2. The network has 10 layers: 1 input layer, 3 convolution layers, 3 dropout layers, 1 max-pooling layer, 1 full connection layer, and 1 output layer.

4.2.1. Input Layer. The first layer is the input layer. After preprocessing, such as data set cleaning and data normalization, the dimension of a single intrusion detection data is changed from 1×43 to 1×128 and the processed data is directly input into the input layer.

4.2.2. Convolutional Layer. Layers 2, 4, and 6 are convolution layers. Convolution layer has the characteristics of sparse connection and parameter sharing. Traditional neural networks use matrix multiplication when establishing the relationship between output and input, which means that each parameter in the parameter matrix participates in describing the interaction between an input unit and an output unit. However, the convolution layer has sparsity, also known as sparse connectivity or sparse interactions.

In terms of convolution calculation, the Gated Convolutional Neural Network (GCNN) is adopted. The gated convolutional method is obtained by the gating idea of recurrent neural network; that is, the useless information is forgotten and the useful information is retained. Based on KDD99 data set, the specific calculation formula of gated convolutional is as follows:

$$\begin{aligned} A &= E \cdot \omega_1 + b_1, \\ B &= E \cdot \omega_2 + b_2, \\ h_l(E) &= A \otimes \mathrm{ReLu}(B), \end{aligned} \tag{3}$$

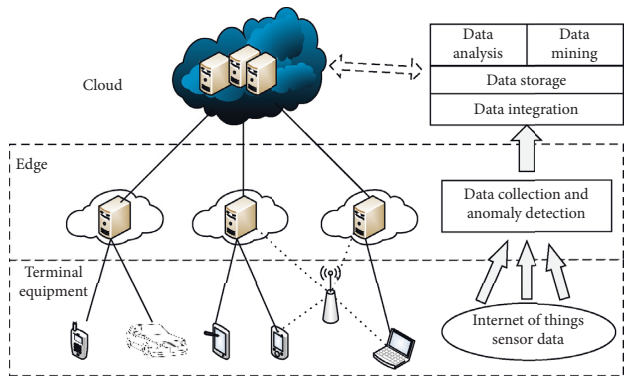

Figure 1: Overall system architecture based on edge computing.

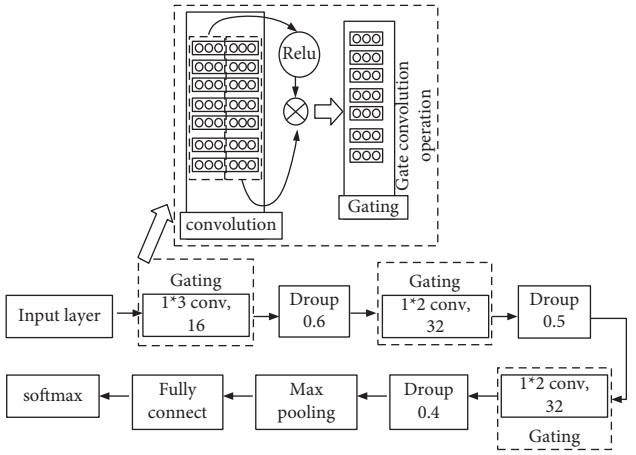

Figure 2: Intrusion detection architecture based on CNN.

where E is the output of the upper layer, ω_1 and ω_2 are the weights, b_1 and b_2 are the offsets, ReLu is the activation function, and $\otimes$ represents the multiplication between the elements in the matrix. Gated convolution is obtained by multiplying the values calculated by two different convolutions. The value of one convolution calculation is directly obtained through linear calculation, that is, A. Another value of convolution calculation is obtained after inputting the linear calculation result into the activation function, that is, B. Then, A and B are multiplied to obtain the gated convolution value.

The proposed network model has three different convolution layers. In the design of convolution kernel size of different convolution layers, the decreasing strategy of convolution kernel size and the increasing strategy of convolution kernel number are adopted. That is, the convolution kernel size is set to 1×3, 1×2, and 1×1 from top to bottom. The number of convolution kernels is set to 16, 32, and 64, respectively. In the shallow convolution layer, a larger convolution kernel is used to extract more local features. In the deep convolution layer, a smaller convolution kernel is used to enhance local features, and smaller convolution kernel can obtain optimal local features and classification performance. In addition, the small convolution kernel can cluster the learned features, which can alleviate the impact of convolution redundancy on the performance of the model to a certain extent.

4.2.3. Dropout Layer. In order to prevent overfitting and improve the generalization ability of model, half of the feature detectors are stopped during each training. This operation is dropout. In short, dropout is to discard some neurons with a certain probability; that is, let the outputs of the discarded neurons be 0. For example, when dropout is set to 0.5, it means that 50% of neurons are discarded and their outputs are 0. Because the labels of the data sets are unbalanced, the CNN model is prone to overfitting in the training process, which has a great impact on the actual classification performance of the model. The dropout method can alleviate the overfitting phenomenon. In order to alleviate the overfitting problem of the proposed model, the dropout layer is adopted in layers 3, 5, and 7 of the model, and the dropout values are set to 0.6, 0.5, and 0.4, respectively, which are set according to the actual situation of the model and the classification effect.

4.2.4. Max-Pooling Layer. Max-pooling layer is the pooling layer using the maximum pooling method to compress the features and remove the redundant features. In addition, the pooling layer also reduces the amount of calculation of the model. The layer 8 of the proposed model is the max-pooling layer, and the stride is 2; that is, the number of parameters is reduced to half of the original.

4.2.5. Fully Connected Layer. Generally, there will be one or more fully connected layers in CNN model. The proposed model adopts 1 full connection layer, and the number of neurons in the fully connected layer is set to 250.

4.2.6. Output Layer. Output layer is a classifier. In deep learning, Softmax function is usually used as a classifier. Softmax classification is a generalization of logistic regression model, which is often used for multiclassification in CNN. During binary classification, Softmax classification will degenerate into logistic regression. In the multiclassification problem, the class label y takes three or more values; that is, the class label y has k ($k > 2$) different values. Given a data set $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, y_i belongs to $\{1, 2, \ldots, k\}$. For a given data x, Softmax estimates the probability of each class in the k class labels. The regression assumption function of Softmax is as follows:

$$g_\vartheta\left(x^{(i)}\right) = \begin{bmatrix} p\left(y^{(i)} = 1 | x^{(i)}; \vartheta\right) \\ p\left(y^{(i)} = 2 | x^{(i)}; \vartheta\right) \\ \vdots \\ p\left(y^{(i)} = k | x^{(i)}; \vartheta\right) \end{bmatrix} = z \cdot \begin{bmatrix} e_1^{\vartheta} x^{(i)} \\ e_2^{\vartheta} x^{(i)} \\ \vdots \\ e_k^{\vartheta} x^{(i)} \end{bmatrix}, \quad (4)$$

$$z = \frac{1}{\sum_{j=1}^{k} e^{\vartheta_j^k x(i)}},$$

where $\vartheta_1, \vartheta_2, \ldots, \vartheta_k$ are the parameters of the model and z represents the normalization of the probability distribution.

In order to facilitate understanding and representation, 1 {.} is used as an indicative function; that is, 1{expression with true value} = 1, 1{expression with false value} = 0. The Softmax cost function is as follows:

$$J(\vartheta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\}\log\frac{e^{\vartheta_j^T x(i)}}{\sum_{j=1}^{k} e^{\vartheta_j^T x(i)}}\right]. \quad (5)$$

The probability of classifying x into category j in Softmax regression is

$$p\left(y^{(i)} = j|x^{(i)}; \vartheta\right) = \frac{e^{\vartheta_j^T x(i)}}{\sum_{j=1}^{k} e^{\vartheta_j^T x(i)}}. \quad (6)$$

The category with the maximum probability is the category of x after Softmax regression.

Softmax classification is usually used as a standard classifier for multiclassification problems. The proposed model uses Softmax classification as a multiclassifier to classify the intrusion detection data. In order to speed up the convergence time of the model under the limited computer resources, Batch Normalization, Adam algorithm, and other methods are used.

4.3. Model Optimization. There are great differences among the labels of intrusion detection data sets. The attack class is a minority class relative to the normal class, and the individual attack class is a minority class relative to the majority of attack classes. Therefore, the loss function is optimized at the algorithm level. The loss function is usually used to show the difference between the actual value and the predicted value for measuring the performance of the model. The smaller the loss function, the better the model. The commonly used loss functions include mean square error and cross entropy. The gradient disappearance problem is not considered in the mean square error. The application of cross entropy includes Sigmoid cross entropy, balanced Sigmoid cross entropy, and Softmax cross entropy, but its result is more obvious on balanced data sets.

Under the binary classification, the cross entropy loss function is

$$\begin{aligned}\mathrm{Loss}(P, y) &= -y\log(P) - (1-y)\log(1-P),\\ &= -\begin{cases} -\log(P), & y = 1,\\ -\log(1-P), & \text{Others},\end{cases}\end{aligned} \quad (7)$$

where $y \in [-1, +1]$ is ground truth class, $y = 1$ is positive class, $y = -1$ is negative class, and $P \in [0, 1]$ is the estimated probability of the model for class $y = 1$.

Let

$$P_t = \begin{cases} P, & y = 1,\\ 1-P, & \text{Others}.\end{cases} \quad (8)$$

Then, cross entropy can be written as

$$\mathrm{Loss}(P, y) = \mathrm{Loss}(P_t) = -\log(P_t). \quad (9)$$

As can be seen from (9), for positive samples, the smaller the loss is, the greater the prediction probability is; for negative samples, the smaller the loss is, the smaller the prediction probability is; there is deviation, and it is not easy to achieve the optimization in the iteration process of positive samples.

In order to avoid serious classification bias in training, it is necessary to reduce the weight of the class with large number samples. The Focal Loss function is improved based on the cross entropy loss function. This function adds modulation factors $(1-P_t)^\tau$ and variable parameters $\tau \in [0, 5]$ based on the cross entropy loss. The parameter τ reduces the loss of easily divided samples of the classes with large number samples, making the classifier pay more attention to difficult divided samples of the classes with few samples. Loss function is written as

$$F\mathrm{Loss}(P, y) = -(1-P_t)^\tau\log(P_t). \quad (10)$$

On this basis, the balance factor $\alpha \in [0, 1]$ is introduced. Variable α determines the importance of positive and negative samples. A definition similar to p_t can be obtained:

$$\alpha_t = \begin{cases} \alpha, & y = 1,\\ 1-\alpha, & \text{Others}.\end{cases} \quad (11)$$

Therefore, the calculation formula of Focal Loss function is

$$\begin{aligned}F\mathrm{Loss}(P_t) &= -\alpha_t(1-P_t)^\tau\log(P_t),\\ &= \begin{cases} -\alpha(1-P_t)^\tau\log(P_t), & y = 1,\\ -(1-\alpha)P_t\log(1-P_t)^\tau, & \text{Others}.\end{cases}\end{aligned} \quad (12)$$

When positive samples $y = 1$ are classified correctly, $P_t = P \geq 0.5$, the modulation factor $(1-P_t)^\tau$ approaches 0. The loss of correctly classified samples reduces the weight. If the classification is wrong, $P_t = P \leq 0.5$, the modulation factor $(1-P_t)^\tau$ approaches 1, so $F\mathrm{Loss}(P_t)$ will not be affected at this time. When negative samples $y = -1$ are classified correctly, $P_t = (1-P) \geq 0.5$, the modulation factor $(1-P_t)^\tau$ approaches 0. The loss of correctly classified samples reduces the weight. If the classification is wrong, $P_t = (1-P) \leq 0.5$, the modulation factor $(1-P_t)^\tau$ approaches 1, so $F\mathrm{Loss}(P_t)$ will not be affected at this time. When the variable parameter τ is equal to 0, the Focal Loss function becomes the cross entropy function. When τ increases, the influence of modulation factor $(1-P_t)^\tau$ increases, reducing the loss of most classes with samples easy to classify. For different values of τ, the change of loss function is shown in Figure 3.

In order to verify the effectiveness of the Focal Loss used in the experiments, several experiments were carried out on α and τ with different values. α is set three values, respectively, 0.2, 0.4, and 0.6, and τ is set four values, respectively, 0, 1, 2, and 5. When α and τ take different values, the performance of the model is shown in Table 1.

As can be seen from Table 1, when $\alpha = 0.4$ and $\tau = 1$, the accuracy is 0.24% higher than that when $\alpha = 0.2$ and $\tau = 2$, but the detection rate is reduced by 0.64% and the false

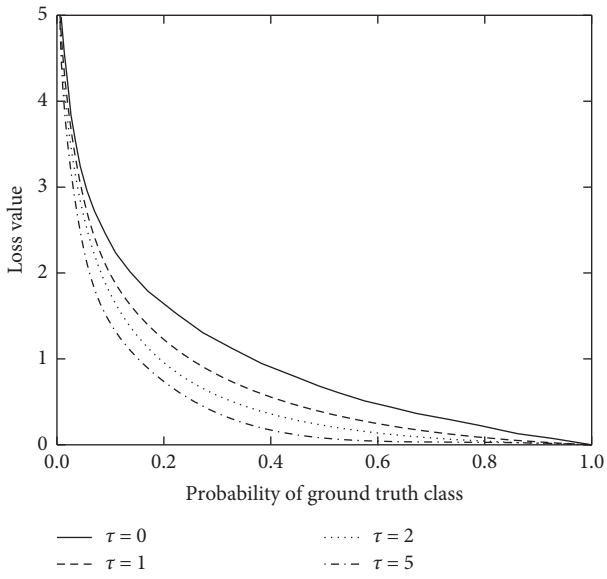

Figure 3: Variation of loss function value with τ value.

positive rate is increased by 0.37%. Therefore, overall, when $\alpha = 0.2$ and $\tau = 2$, the performance of the model is the best, so $\alpha = 0.2$ and $\tau = 2$ are used in the proposed model. According to the above analysis, Focal Loss solves the problems of imbalance between positive and negative samples and distinguishing simple and complex samples. Therefore, the Focal Loss is adopted to optimize the proposed intrusion detection model.

5. Experiment and Analysis

In order to evaluate the performance of the proposed algorithm, Python version 3.7.10 and Keras version 2.3.1 were used, and TensorFlow was used as the back end. A compatible computer is used for simulation experiments. The specific configurations are i7-6700_3.40 GHZ, DDR4_8G_2133, and Win10 version LSTC2019.

Before the final experiment, it is necessary to optimize some parameters. A large number of experiments were conducted based on predecessors to set each parameter and determine the final parameters. The learning rate is 0.0001, the maximum number of iterations is 200, and each batch of training is 256. In the optimization algorithm Adam, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. In Focal Loss, $\alpha = 0.2$ and $\gamma = 0.2$. The activation function is ReLu.

5.1. Evaluation Index. When using balanced data sets, the evaluation index of the algorithm is often accuracy. In the unbalanced data set, the unprocessed classifier is prone to the large number of samples. Although the accuracy will obtain a higher value, this is based on the misclassification of a few classes. For example, there are 98 positive examples and 2 negative examples. When the classification algorithm predicts all samples as positive examples, it can also achieve 98% accuracy, but it does not predict any negative examples, so such a classifier is obviously worthless. Therefore, in the unbalanced data set, the evaluation index should not only use the accuracy, but also consider the precision, recall, and $F1$ score. Due to the imbalance of the KDD99 data set, the evaluation indicators not only use the accuracy, but also use the precision, recall, $F1$ score, and other indicators.

Accuracy refers to the proportion of the number of correctly classified samples to the total number of samples. For set $\psi = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, accuracy is defined as follows:

$$\mathrm{ACC}(f; \psi) = \frac{1}{m} \sum_{i=1}^{m} \prod (f(x_i) = y_i). \tag{13}$$

According to the combination of the real value of the data and the predicted value of the classifier, it can be divided into four results: true positive (TP), both real and predicted values are positive; false positive (FP): the data with negative value are predicted as positive; true negative (TN): both real and predicted are negative; false negative (FN): the real value is positive but the prediction is negative. The confusion matrix can be obtained according to the four results. On this basis, the precision and recall are proposed. Precision represents the proportion of the true positive cases among all predicted positive cases. Recall indicates the integrity of the prediction of positive cases, that is, the proportion of true positive cases among all real positive cases. The specific calculation of the two is as follows:

$$\begin{aligned} \mathrm{Precision} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \\ \mathrm{Recall} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \end{aligned} \tag{14}$$

In general, precision and recall are a pair of contradictory measurements; that is, they show an inverse relationship. When the precision is improved, the recall rate will be reduced, and when the recall rate is improved, the precision will be reduced. F-Measure integrates the precision and recall, which is a typical trade-off function. F-Measure will improve only when precision and recall are improved at the same time. The formula of F-Measure is shown in the following equation. Only when the value of F-Measure is high, can it indicate that the classifier has good performance:

$$F\text{-measure} = \frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}. \tag{15}$$

5.2. Changes of Precision and Loss Rate in Iterative Training. The samples of the proposed algorithm can be divided into five classes according to the class identification. The iterative rounds of algorithm training are set to 200. The change curves of precision and loss rate of training set and testing set during the training process are shown in Figure 4.

As can be seen from Figure 4, when the iteration rounds reach 20 times, the changes of precision and loss rate of the proposed algorithm tend to be stable. In the training process, the classification precision of attack samples and normal samples reaches 99.7%, and the precision tends to be stable. The average precision of the testing set reaches 99.6%. At the

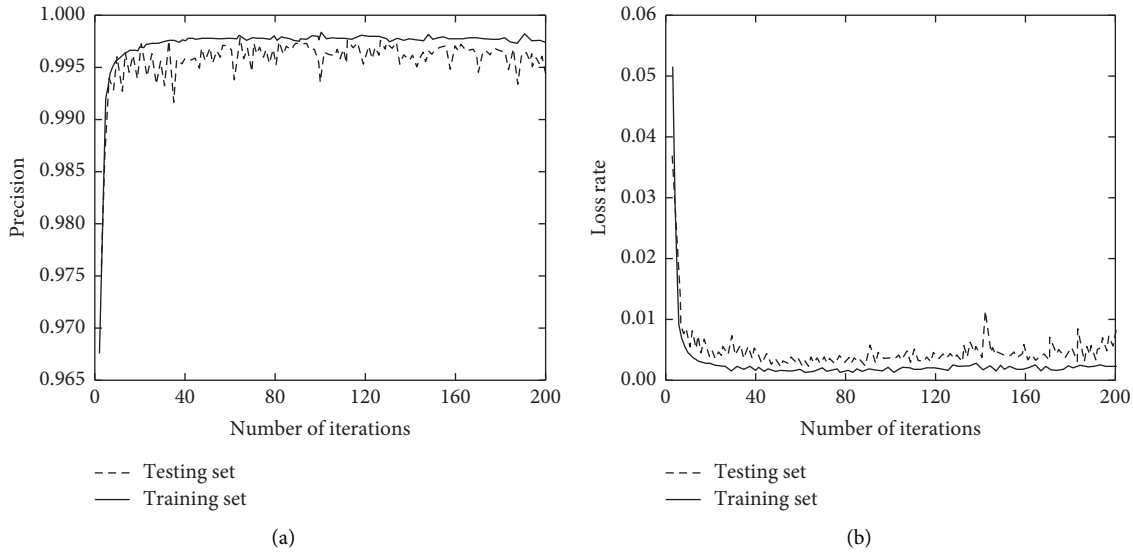

Figure 4: Changes in precision and loss rate. (a) Precision change. (b) Loss rate change.

same time, the average loss of the test set is less than 0.005, which can demonstrate that the proposed intrusion detection algorithm has good classification ability for KDD99 data set. In order to reduce the calculation amount of CNN model and ensure the detection accuracy, gated convolution is introduced into CNN model, which forgets useless information and retains useful information. IoT data is classified through convolution, pooling, dropout, full connection, and Softmax function, and the Focal Loss function is used to solve the problem of uneven distribution of sample data.

5.3. Comparison of Time Spent in Each Stage. According to different data processing methods, the corresponding average processing time is also different. Among them, the received data scale of cloud computing is large, so the bandwidth pressure is high. Network transmission time of cloud computing is significantly longer than that of edge computing. The edge computing architecture can reduce the delay and save time. The time consumption of different stages is shown in Figure 5.

As can be seen from Figure 5, the computing power of the cloud central node is relatively strong, but, under the pressure of data scale and bandwidth, the processing time of cloud computing is relatively high, exceeding 280 ms in the receiving and storage stages. In the data detection based on edge computing, because it completes the data processing task "nearby," the time consumptions are relatively short, which are 236 ms and 157 ms in the stages of receiving and storing data, respectively. Therefore, through comprehensive comparison and analysis, the anomaly detection performance of edge computing is better.

5.4. Comparison of Overall Indices with Other Methods. In order to demonstrate the performance of the proposed algorithm, it is compared with [11, 14, 20]. The results of overall indices are shown in Figure 6.

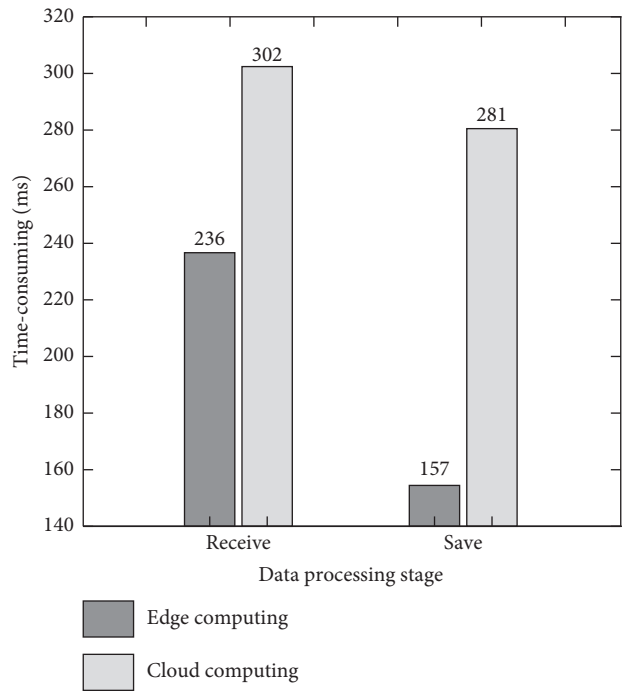

Figure 5: Time consuming comparison of different stages of data processing.

As can be seen from Figure 6, compared with other algorithms, the overall performance of the proposed algorithm is the best. The accuracy, precision, recall, and $F1$ values are 92.14%, 95.97%, 90.89%, and 90.03%, respectively. Because the proposed algorithm uses the improved CNN model to realize data type classification and uses the Focal Loss function to optimize the algorithm, the classification error is reduced and the accuracy of intrusion detection is improved. Reference [11] uses singular value decomposition technology for feature dimensionality reduction, and reconstruction loss is used to determine the intrusion type of a

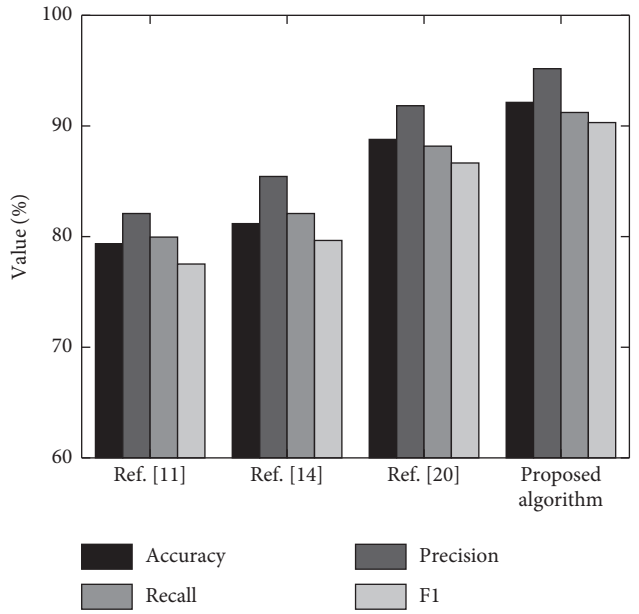

Figure 6: Overall index comparison results of different algorithms.

given network feature, but it is difficult to deal with complex and changeable network intrusion. Therefore, the overall performance is poor. The $F1$ value is less than 78%. Reference [14] combines the efficient clustering technology based on adaptive chicken swarm optimization algorithm and SVM classification model to detect malicious sensor nodes. This algorithm realizes intrusion detection and improves the detection performance to a certain extent. However, due to the large amount of IoT data and various types of network intrusion, the detection accuracy is only 82.37%. Reference [20] uses a new feature selection method based on conditional random field and linear correlation coefficient and uses the traditional CNN network to complete sample classification. Because there is no problem of large difference in label items in the data set, its precision is higher than 90%, but its recall rate is only 87.65%. The overall performance needs to be improved.

5.5. Comparison of Precision and Recall. The KDD99 data set has five class types of intrusion data; the precision and recall results obtained by the four algorithms for each type are shown in Tables 2 and 3.

It can be seen from Tables 2 and 3 that the precision and recall of the proposed algorithm are better than the comparison algorithms in five categories. Taking Normal as an example, the precision and recall are 78.70% and 96.52%, respectively, which is 5.95% and 2.58% higher than the detection algorithm using traditional CNN in [20]. In addition, the classification results of the four algorithms for U2R are not ideal, and the precision and recall rates are lower than 74% and 25%, respectively. Because this label type is an unauthorized user, it is difficult to be identified, so the detection effect is poor. On the contrary, there are many researches on DoS type, and the intrusion detection technology is relatively mature, so the precision of the proposed

Table 2: Comparison results of precision of different algorithms.

Algorithm	Precision/%				
	Normal	Probe	DoS	U2R	R2L
The proposed algorithm	78.70	87.44	96.52	73.80	92.94
Ref. [20]	72.75	75.99	93.57	71.80	92.59
Ref. [14]	70.39	73.30	90.34	63.62	89.07
Ref. [11]	63.54	74.56	83.45	68.76	86.75

Table 3: Comparison results of recall of different algorithms.

Algorithm	Recall/%				
	Normal	Probe	DoS	U2R	R2L
The proposed algorithm	96.52	75.66	86.45	24.40	45.54
Ref. [20]	93.94	70.60	80.10	13.95	29.94
Ref. [14]	90.37	68.71	80.34	11.91	23.86
Ref. [11]	87.56	65.68	75.65	9.24	19.85

algorithm is as high as 96.52%, and the recall rate is 86.45%. In general, the proposed algorithm improves the CNN model, and the Focal Loss is effective.

6. Conclusion

With the rapid development of the Internet and the construction of new generation of information infrastructure, IoT intrusion behavior is becoming more and more common. How to ensure IoT security is one of the current research hotspots. Therefore, an IoT intrusion detection algorithm based on deep learning in edge computing environment is proposed. The preprocessed KDD99 data set is input into the intrusion detection model based on edge calculation, the improved CNN model is used to realize data multiclassification, and the Focal Loss function is used to modulate the training ratio of positive and negative samples. The gated convolution is introduced into the improved CNN model to simplify the model and ensure the detection accuracy. The accuracy, precision, recall, and $F1$ values are 92.14%, 95.97%, 90.89%, and 90.03%, respectively, which are better than other comparison algorithms and provide a new solution for the field of intrusion detection.

At present, the field of network security is based on the combination of various methods, and there is no perfect depth-learning model. Therefore, the selection of parameters in the process of model training and optimization is mostly obtained by reference papers and experimental experience, which lacks theoretical guidance and persuasion. Next, the theoretical and practical guidance of depth learning in the field of intrusion detection should continue to be studied.

Data Availability

The data used to support the findings of this study are included within the 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 Guangdong Characteristic Innovation Project (2021KTSCX159), Special Projects in Key Fields of Colleges and Universities in Guangdong Province (2021ZDZX1070), project supported by the Industry University Research Innovation Fund of Chinese Universities (2021FNA04013), and School Level Scientific Research Project of Guangzhou Institute of Science and Technology (2021KZ002).

References

[1] A. Jain, T. Singh, and S. Kumar Sharma, "Security as a solution: an intrusion detection system using a neural network for IoT enabled healthcare ecosystem," *Interdisciplinary Journal of Information, Knowledge, and Management*, vol. 16, no. 5, pp. 331–369, 2021.

[2] L. Zhu, "Safety detection algorithm in sensor network based on ant colony optimization with improved multiple clustering algorithms," *Safety Science*, vol. 118, pp. 96–102, 2019.

[3] 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 Transactions on Emerging Topics in Computing*, vol. 7, no. 2, pp. 314–323, 2019.

[4] S. Dwivedi, M. Vardhan, and S. Tripathi, "Distributed denial-of-service prediction on IoT framework by learning techniques," *Open Computer Science*, vol. 10, no. 1, pp. 220–230, 2020.

[5] C. Wu, Y. Liu, F. Wu et al., "A hybrid intrusion detection system for IoT applications with constrained resources," *International Journal of Digital Crime and Forensics*, vol. 12, no. 1, pp. 109–130, 2020.

[6] D. Li and K. Yang, "A dual-port 8-T CAM-based network intrusion detection engine for IoT," *IEEE Solid-State Circuits Letters*, vol. 3, no. 6, pp. 358–361, 2020.

[7] R. A. Ramadan and K. Yadav, "A novel hybrid intrusion detection system (IDS) for the detection of Internet of things (IoT) network attacks," *Annals of Emerging Technologies in Computing*, vol. 4, no. 5, pp. 61–74, 2020.

[8] H. B. Patel and D. C. Jinwala, "6MID:Mircochain based intrusion detection for 6LoWPAN based IoT networks," *Procedia Computer Science*, vol. 184, no. 6, pp. 929–934, 2021.

[9] P. Kumar, G. P. Gupta, and R. Tripathi, "Design of anomaly-based intrusion detection system using fog computing for IoT network," *Automatic Control and Computer Sciences*, vol. 55, no. 2, pp. 137–147, 2021.

[10] H. Al-Hamadi, I. R. Chen, D. C. Wang, and M. Almashan, "Attack and defense strategies for intrusion detection in autonomous distributed IoT systems," *IEEE Access*, vol. 8, no. 8, Article ID 168994, 2020.

[11] Y. Alagrash, A. Drebee, and N. Zirjawi, "Comparing the area of data mining algorithms in network intrusion detection," *Journal of Information Security*, vol. 11, pp. 1–18, 2020.

[12] T. R. Ahmad, L. M. Chen, J. S. Chapman, and L. L. Chen, "Medicaid and Medicare payer status are associated with worse surgical outcomes in gynecologic oncology," *Gynecologic Oncology*, vol. 155, no. 1, pp. 93–97, 2019.

[13] A. J. Siddiqui and A. Boukerche, "TempoCode-IoT: temporal codebook-based encoding of flow features for intrusion detection in Internet of Things," *Cluster Computing*, vol. 24, no. 1, pp. 17–35, 2021.

[14] G. M. Borkar, L. H. Patil, D. Dalgade, and A. Hutke, "A novel clustering approach and adaptive SVM classifier for intrusion detection in WSN: a data mining concept," *Sustainable Computing: Informatics and Systems*, vol. 23, no. 9, pp. 120–135, 2019.

[15] T. Mawla, S. Dutta, and M. F. Rabbi, "Temporal signature mining for network intrusion detection using TEMR," *Advances in Intelligent Systems and Computing*, vol. 10, no. 57, pp. 645–655, 2019.

[16] S. Smys, A. Basar, and H. Wang, "Hybrid intrusion detection system for Internet of things (IoT)," *Journal of ISMAC*, vol. 2, no. 4, pp. 190–199, 2020.

[17] S. Gavel, A. S. Raghuvanshi, and S. Tiwari, "Distributed intrusion detection scheme using dual-axis dimensionality reduction for Internet of things (IoT)," *The Journal of Supercomputing*, vol. 77, no. 2, pp. 1–24, 2021.

[18] N. Islam, F. Farhin, I. Sultana et al., "Towards machine learning based intrusion detection in IoT networks," *Computers, Materials & Continua*, vol. 69, no. 2, pp. 1801–1821, 2021.

[19] A. Azawii, S. Al-Janabi, and B. Al-Khateeb, "Survey on intrusion detection systems based on deep learning," *Periodicals of Engineering and Natural Sciences*, vol. 7, no. 3, pp. 1074–1095, 2019.

[20] B. Riyaz and S. Ganapathy, "A deep learning approach for effective intrusion detection in wireless networks using CNN," *Soft Computing*, vol. 24, no. 22, Article ID 17265, 2020.

[21] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, no. 2, Article ID 102419, 2020.

[22] J. D. Lee, H. S. Cha, S. Rathore, and H P. Jong, "M-IDM A multi-classification based intrusion detection model in healthcare IoT," *Computers, Materials & Continua*, vol. 67, no. 2, pp. 1537–1553, 2021.

[23] S. Fenanir, F. Semchedine, S. Harous, and A. Baadache, "A semi-supervised deep auto-encoder based intrusion detection for IoT," *Ingénierie des Systèmes d'Information*, vol. 25, no. 5, pp. 569–577, 2020.

[24] N. Xiaomei, W. Huawei, C. Changchang, H. Jiyu, and S. Zhongdong, "Civil aviation safety evaluation based on deep belief network and principal component analysis - ScienceDirect," *Safety Science*, vol. 112, pp. 90–95, 2019.

[25] O. Faust, E. J. Ciaccio, A. Majid, and U. R. Acharya, "Improving the safety of atrial fibrillation monitoring systems through human verification," *Safety Science*, vol. 118, no. 118, pp. 881–886, 2019.

[26] Z. Fan, C. Liu, D. Cai, and S. Yue, "Research on black spot identification of safety in urban traffic accidents based on machine learning method," *Safety Science*, vol. 118, no. 118, pp. 607–616, 2019.