Hindawi
International Journal of Distributed Sensor Networks
Volume 2023, Article ID 1982173, 12 pages
https://doi.org/10.1155/2023/1982173

Research Article

An Intrusion Detection Model Based on Feature Selection and Improved One-Dimensional Convolutional Neural Network

Qingfeng Li,[1] **Bo Li**,[2] **and Linzhi Wen**[2]

[1]*Network Information Center, Northeast Forestry University, Harbin, Heilongjiang 150040, China*
[2]*College of Information and Computer Engineering, Northeast Forestry University, Harbin, Heilongjiang 150040, China*

Correspondence should be addressed to Bo Li; 2020111884@nefu.edu.cn

Received 26 September 2023; Revised 26 November 2023; Accepted 2 December 2023; Published 21 December 2023

Academic Editor: José Molina

The problem of intrusion detection has new solutions, thanks to the widespread use of machine learning in the field of network security, but it still has a few issues at this time. Traditional machine learning techniques to intrusion detection rely on expert experience to choose features, and deep learning approaches have a low detection efficiency. In this paper, an intrusion detection model based on feature selection and improved one-dimensional convolutional neural network was proposed. This model first used the extreme gradient boosting decision tree (XGboost) algorithm to sort the preprocessed data, and then it used comparison to weed out 55 features with a higher contribution. Then, the extracted features were fed into the improved one-dimensional convolutional neural network (I1DCNN), and this network training was used to complete the final classification task. The feature selection and improved one-dimensional convolutional neural network (FS-I1DCNN) intrusion detection model not only solved the traditional machine learning method of relying on expert experience to extract features but also improved the detection efficiency of the model, reduced the training time while reducing the dimension, and increased the overall accuracy. In comparison to the I1DCNN model without feature extraction and the conventional one-dimensional convolutional neural network (1DCNN) model, the experimental results demonstrate that the FS-I1DCNN model's overall accuracy increases by 0.67% and 2.94%, respectively. Its accuracy, precision, recall, and $F1$-score were significantly better than those of the other intrusion detection models, including SVM and DBN.

1. Introduction

The Internet has become one of the key tools that we cannot live without thanks to the advancement of science and technology, but in recent years, the complexity of the network environment has led to an increasing number of network security incidents around the world [1, 2], and the number of network attacks on various countries is increasing. Therefore, the research on intrusion detection technology has become an indispensable link in the field of network security research. Intrusion detection technology became a dynamic and crucial security protection tool, opening the second line of defense after firewall [3]. It has since developed into a crucial way to defend against network intrusion in the current era of widely used encrypted traffic [4].

Different intrusion detection models based on machine learning and deep learning are being improved more and more as artificial intelligence technology advances quickly [5, 6], but both have disadvantages. SVM, decision trees, and other common algorithms [7–9] are used frequently in traditional machine learning. Their main issue is that they frequently lose sight of the connections between features when extracting features because they rely too heavily on expert experience. Wang et al. [10] proposed a support vector machine based intrusion detection framework. They implement the logarithm marginal density ratios transformation to form new transformed features. In this way, they improve the capability of SVM detection model. In order to reduce the complexity of the model, Kim et al. [11] established a hierarchically integrated anomaly detection model that employs decision trees to build misuse models and data decomposition to create smaller subsets and single-class SVM models from the subsets. Both of the above use SVM to build detection models; however, the training speed drops

dramatically when the training data increases substantially [12], so the SVM model is not a very good choice.

Deep learning has been used extensively in the field of intrusion detection recently, but its issue has been poor detection performance. Wang et al. [13] proposed an intrusion detection method based on feature optimization and BP neural network, which improves the intrusion detection rate of a few categories while reducing the dimensionality. However, since BP neural network has a large number of parameters, the convergence speed is relatively slow, leading to low training efficiency. In order to effectively reduce feature dimensionality while maintaining detection performance, Luo and Lu [14] developed a hybrid network attack detection algorithm based on artificial neural networks and genetic algorithms. However, the genetic algorithm itself necessitates the process of encoding and decoding, resulting in a lengthy training period. In the field of intrusion detection, deep neural networks and convolutional neural networks significantly outperform BP neural networks in terms of training efficiency, but the accuracy rate still needs to be increased. Zhang et al. [15] established a deep convolutional neural network classification model based on the improved PCA algorithm, which improves the accuracy of detection while using the PCA algorithm for dimensionality reduction, but the overall accuracy is relatively low. Yang and Wang [16] proposed an improved convolutional neural network intrusion detection method that abstracts low-level traffic into high-level features. The low-level intrusion traffic data is abstractly represented as advanced features by CNN. The stochastic gradient descent algorithm is used to converge the model and the optimization algorithm is used for parameter tuning, and better results are achieved. Moreover, compared with the deep convolutional neural network with too high dimensionality, the one-dimensional convolutional neural network is not only low-dimensional but also a better choice for intrusion detection data with high chronology. Qazi et al. [17] proposed a one-dimensional convolutional neural network-based deep learning system for network intrusion detection and achieved an accuracy rate of 98.96%. However, this system only detects four categories of attacks and is not truly pervasive. Hang et al. [18] proposed an improved method of a one-dimensional convolutional neural network, which used the results of two convolutions as the input of global average pooling and global maximum pooling, and combined the input data to improve the network intrusion detection rate and reduce the parameters and training time of the model.

In summary, to address the problems in intrusion detection, this paper, based on existing research, designs a feature selection and improved one-dimensional convolutional neural network (FS-I1DCNN) intrusion detection model. The key contributions of the paper are as follows:

(1) We used the methods of oversampling, undersampling, and mean square normalization to process the original dataset

(2) We adopted XGboost feature selection method to select and filter the processed data, which reduced the training time of the model and sped up the operation efficiency

(3) We designed an improved one-dimensional convolutional neural network (I1DCNN) intrusion detection model and compared different optimization algorithms. The Adam optimization algorithm is finally used to adjust the model parameters dynamically

The main distribution of this article is as follows. Section 2 introduces the research methods of this article. Furthermore, we provide the experimental results and analysis in Section 3. Finally, we draw research conclusions and prospects.

2. Related Knowledge

2.1. XGboost Feature Selection Algorithm. The XGboost (extreme gradient boosting) algorithm [19] is an evolution of the GDBT (gradient boosting) algorithm and is an efficient system implementation. In this paper, we make use of its tree model to quantify it and choose the features based on their relative importance.

2.1.1. Decision Tree Model and Their Combinations. In the decision tree model construction process, the feature segmentation points are greedily selected using each layer so that they are used as leaf nodes, and then the entire tree is made to gain the most. This means that the more times the feature is segmented, the greater the average gain the feature brings to the whole tree, which indicates that the feature is more important compared to other features. The weight of each leaf node during segmentation can be expressed as $w(g_i, h_i)$; g_i and h_i are displayed in equations (1) and (2), respectively.

$$g_i = \partial_{y^{(t-1)}} l\left(y_i, \widehat{y}^{(t-1)}\right), \tag{1}$$

$$h_i = \partial^2_{\widehat{y}^{(t-1)}} l\left(y_i, \widehat{y}^{(t-1)}\right). \tag{2}$$

The training error $l(y_i, \widehat{y}_i)$ is the difference between the training target value y_i and the predicted value $\widehat{y}_i$. The gain of each feature as a segmentation point is shown in equation (3), which means that for each segmentation point, the gain can be expressed as the difference between the total weights after segmentation and the total weights of the leaf nodes before segmentation, where the total weight is the sum of the total weights of the left and right subtrees. This would be made in order to minimize the cost of the segmented tree.

$$\text{Gain} = \sum_{\text{left}} w + \sum_{\text{right}} w - \sum_{\text{nosplit}} w. \tag{3}$$

By continuously iteratively creating new trees, compiling all the trees into a final result, adding a tree during each iteration [20], and building a linear combination of K trees

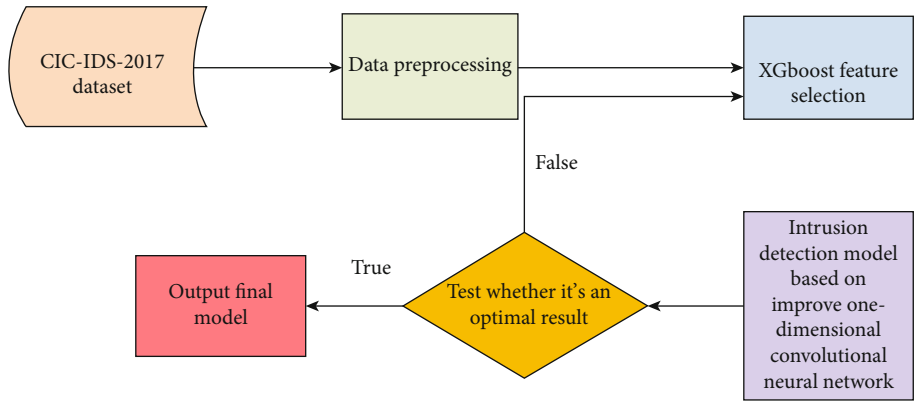

Figure 1: FS-I1DCNN model diagram.

as shown in Equation (4), the XGboost model essentially learns the residuals of the true values and the current predicted values of all trees.

$$\hat{y}^{(t)} = \sum_{k=1}^{K} f_k(x_i) = \hat{y}^{(t-1)} + f_t(x_i), f_k \in F. \tag{4}$$

$f_k(x_i)$ refers to the weight of the leaf node that the kth tree in the sample is categorized into, and F denotes the function space of all trees.

2.1.2. Importance Metrics. The importance metric is a measure to evaluate the importance of each feature and refers to the importance score of each attribute obtained by the XGboost algorithm [21]. In order to correctly complete the classification task, XGboost algorithm bases the construction of a decision tree on the number of feature splits *F*Score, the average feature gain value AverageGain, and the average feature coverage AverageCover, as shown in the following equations:

$$F\text{Score} = |X|, \tag{5}$$

$$\text{AverageGain} = \frac{\sum \text{Gain}_X}{F\text{Score}}, \tag{6}$$

$$\text{AverageCover} = \frac{\sum \text{Cover}_X}{F\text{Score}}. \tag{7}$$

As determined by equation (3), X is the set of features assigned to the leaf nodes, Gain is the gain of the$X$th feature for the segmentation point, and Cover denotes the number of samples at each node.

2.2. Convolutional Neural Networks. Convolutional neural network (CNN) is one of the representative deep learning networks. It has been successfully applied to a variety of artificial intelligence applications [22], including computer vision and natural language processing [23, 24]. Compared to conventional intelligent algorithms, CNN has much more powerful feature extraction capabilities, with its main components being the convolutional layer, pooling layer, fully connected layer, and softmax layer.

Table 1: Traffic types and distribution of CIC-IDS-2017.

Data categories	The amount of processed data
Benign	285324
Bot	7864
DDos	128027
Dos GoldenEye	10293
Dos Hulk	231073
Dos slowhttptest	5499
Dos slowloris	5796
FTP-Patator	7938
Heartbleed	5632
Infiltration	4608
PortScan	158930
SSH-Parator	5897
Web Attack	16620

2.2.1. Convolutional Layer. The convolutional layer, which serves as the brain of a convolutional neural network, primarily performs convolution and excitation operations. By swiping the convolution kernel window across the input data, the local regions of the input data are convolved with the convolution kernel. The mathematical expression is shown in the following equation:

$$Z_i^l = f\left(\mathrm{W}_i^{l*} K^{l-1} + \mathrm{b}_i^l\right). \tag{8}$$

l denotes the lth convolutional layer; Z_i^l means the lth feature of the output of the ith convolutional layer; $*$ stands for the convolutional operation; b_i^l shows the bias of the ith convolutional kernel; K^{l-1} indicates the output of the $l-1$th layer; $f()$ and W_i^l denote the activation function and the weight matrix of the ith convolutional kernel of the lth layer, respectively. Relu, sigmoid, tanh, and other frequently used activation functions are included in CNN. The Relu function has the advantages of speeding up convergence, increasing accuracy, and reducing overfitting when compared to other

Input: Intrusion detection dataset S, feature $F = \{t_1, t_2 \cdots t_n\}$, n is feature sum.
Step1: Calculate the importance of each feature, in order from smallest to largest, get $t_1{}', t_2{}' \cdots t_k{}'$, and $k \le n$, and satisfy the condition of $t_k{}' \ge t_{k-1}{}' \ge t_{k-2}{}' \cdots \ge t_1{}'$.
Step2: $Acc = 0$, $i = k$ // Acc is accuracy, and i is the number of retained features
Step3: for i to 1, *step* −1 *do*
Step4: Input the CNN, and record the accuracy obtained as $Acc(i)$
Step5: if $Acc(i)>Acc$ *then*
Step6 $Acc \longleftarrow Acc(i)$
Step7 *End* if
Step8 *End* for
Output: the feature filtered dataset S'

Algorithm 1: The XGboost feature selection algorithm.

functions. In this paper, Relu is chosen as the activation function; its formula is provided in the following:

$$\mathrm{Re\,lu}(x) = \begin{cases} x(x>0) \\ 0(x<0) \end{cases}. \tag{9}$$

2.2.2. Pooling Layer. The primary function of the pooling layer is spatial merging, which is also known as downsampling or subsampling. It basically makes sure that the crucial data is simultaneously reduced in dimensionality. As shown in Equation (10), the maximum pooling layer is used in this study. $Y_i^{l+1}(\mathrm{j})$ stands for the ith feature element of the $l+1$th layer after pooling; D_j denotes the jth pooling region; and $Z_i^j(\mathrm{k})$ means the element of the ith feature map of the lth layer that is included in the scope of this pooling kernel.

$$Y_i^{l+1}(\mathrm{j}) = \max Z_i^j(\mathrm{k}), \mathrm{k} \in D_j. \tag{10}$$

2.2.3. Fully Connected Layer. The final classification result, whose expression is shown in (11), is obtained by combining the previously extracted features through the fully connected layer of a multilayer perceptron to perform nonlinear activation and output the probability distribution of each classification. $P(Y_j)$ is the probability output of the neuron following the softmax activation function, and m denotes the number of classifications.

$$P\left(Y_j\right) = \frac{\exp\left(Y_j\right)}{\sum_{k=1}^{m} \exp(Y_k)} \tag{11}$$

3. Intrusion Detection Model Based on FS-I1DCNN

The FS-I1DCNN intrusion detection model that is proposed in this paper has three main modules, with the specific framework shown in Figure 1. Data preprocessing module, which first samples, filters, and cleans the original dataset; XGboost feature screening module, which prioritizes data by XGboost model and filters out the features with higher contribution to the model by experimental comparison; I1DCNN traffic detection module, which completes the classification task by an improved one-dimensional convolutional neural network [22], and after experimental validation following comparison tests, the best option is chosen, and the FS-I1DCNN intrusion detection model is finished being built.

3.1. Data Preprocessing. The FS-I1DCNN intrusion detection model that is proposed in this paper has three main modules, with the specific framework shown in Figure 1. The first module is data preprocessing module, which includes sampling, filtering and cleaning the original dataset. The second module is XGboost feature screening module, which prioritizes data by XGboost model and filters out the features with higher contribution to the model by experimental comparison. The last module is I1DCNN traffic detection module, which completes the classification task by an improved one-dimensional convolutional neural network [22], and the best scheme is selected after experimental verification. The Canadian Institute for Cybersecurity Research's CIC-IDS-2017 dataset [25] is used in this study. It was collected from 9 a.m. on July 3, 2017, to 5 p.m. on July 7, 2017, and primarily includes 12 categories of attacks and regular benign traffic. Since the dataset has some missing data and is large and has enough experimental data, it should be processed for missing values. The method used is to remove the missing values to stop the missing values from having an effect on the experiment. Furthermore, this experimental dataset contains 78 pertinent features, each of which has a magnitude and order of magnitude that varies. Some of these feature values are of a large order of magnitude, which will affect how well the model performs if trained directly. The feature flow duration, for instance, has a range of feature values between -1 and 119999993, so this feature should be normalized to speed up computing and also remove the impact on the experimental results if the magnitudes are different.

In this study, we employ the demean normalization technique, which preprocesses the data and arranges it uniformly using the StandardScaler module of the sklearn package. The mean-variance normalization formula is displayed in the following:

$$G = \frac{L - L\mathrm{mean}}{\alpha}, \tag{12}$$

where L represents the feature value of each group, Lmean

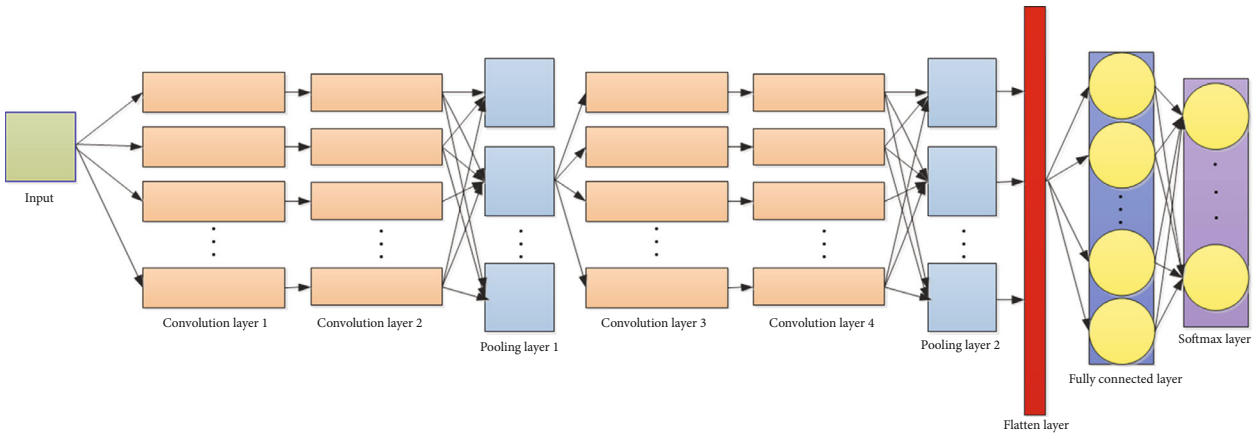

Figure 2: Structure diagram of improved one-dimensional convolutional neural network.

Input: Initial parameter θ and step size ε.Exponential decay rate of moment estimation ρ_1 and ρ_2. They denote updating the momentum term and RMSprop, respectively. The feature filtered dataset S' and minibatch of the training set of u samples $\{x^{(1)}, \cdots x^{(u)}\}$ in the filtered dataset S',whose corresponding target is $y^{(i)}$.
Step1: int $s = 0$, $r = 0$ $t = 0$ // s is a first-order matrix variable, r is a second-order matrix variable, t is a time step
Step2: while failure to meet stop guidelines do
Step3: $g \longleftarrow 1/m\nabla_\theta \sum_i L(f(\mathrm{x}^{(i)}; \theta), \mathrm{y}^i)$ // Calculate gradient
Step4: $t \longleftarrow t + 1$ // Update training times
Step5: $s \longleftarrow \rho_1 s + (1 - \rho_1)\mathrm{g}$ // Cumulative gradient
Step6: $r \longleftarrow \rho_2 r + (1 - \rho_2)\mathrm{g}^2$ // Calculate gradient squared
Step7 $s^\wedge \longleftarrow s/1 - \rho_1^t$ // Correct the deviation of the first moment
Step8 $r^\wedge \longleftarrow r/1 - \rho_2^t$ // Correct the deviation of the second moment
Step9 $\Delta\theta = -\epsilon s^\wedge / \sqrt{r^\wedge} + \delta$ // calculate update, element-by-element operation.
Step10 $\theta \longleftarrow \theta + \Delta\theta$ // Update parameter
Step11 End while
Output: The updated parameter θ'

Algorithm 2: The Adam optimization algorithm.

Table 2: Confusion matrix.

Label		Predict class	
		Positive	Negative
True class	Positive	TP	FN
	Negative	FP	TN

represents the mean value of each group, and α represents the standard deviation of each group.

Each group's feature value, mean value, and standard deviation are represented by the letters L, Lmean, and α, respectively.

The final data categories and quantities obtained following the above data preprocessing are shown in Table 1.

3.2. XGboost Feature Selection. If a feature is being screened, whether it plays a crucial part in the model will determine whether to keep it or not. To achieve the best classification effect, generate the corresponding feature contribution degree based on the XGboost feature importance index mentioned above, sort them based on the size of the feature contribution degree, and run experiments using various retained features. Algorithm 1 illustrates the XGboost feature selection algorithm suggested in this paper.

After the above XGboost selection, the dataset S' is obtained, which has a smaller dimension than the dataset S before the selection, which reduces the time for the following training and speeds up the operation efficiency.

3.3. Improved One-Dimensional Convolutional Neural Network. Convolution-pooling-full connections make up the bulk of traditional convolutional neural networks. Compared with the fully connected neural network, CNNs have fewer parameters when the same number of hidden units is used [26]. Moreover, CNN is easy to train [27]. And one-dimensional convolutional neural networks are a kind of convolutional neural network that can be effectively identified and applied to the time series problem of sensor data or by fixed-length periodic signal data. So this paper selects a one-dimensional convolutional neural network as the core of the classification model according to the characteristics of the temporal sequence of the intrusion detection dataset. In addition, due to the high dimensionality of the intrusion detection dataset, only one layer of convolutional operation does not fully extract the features in the dataset, so in this paper, we design

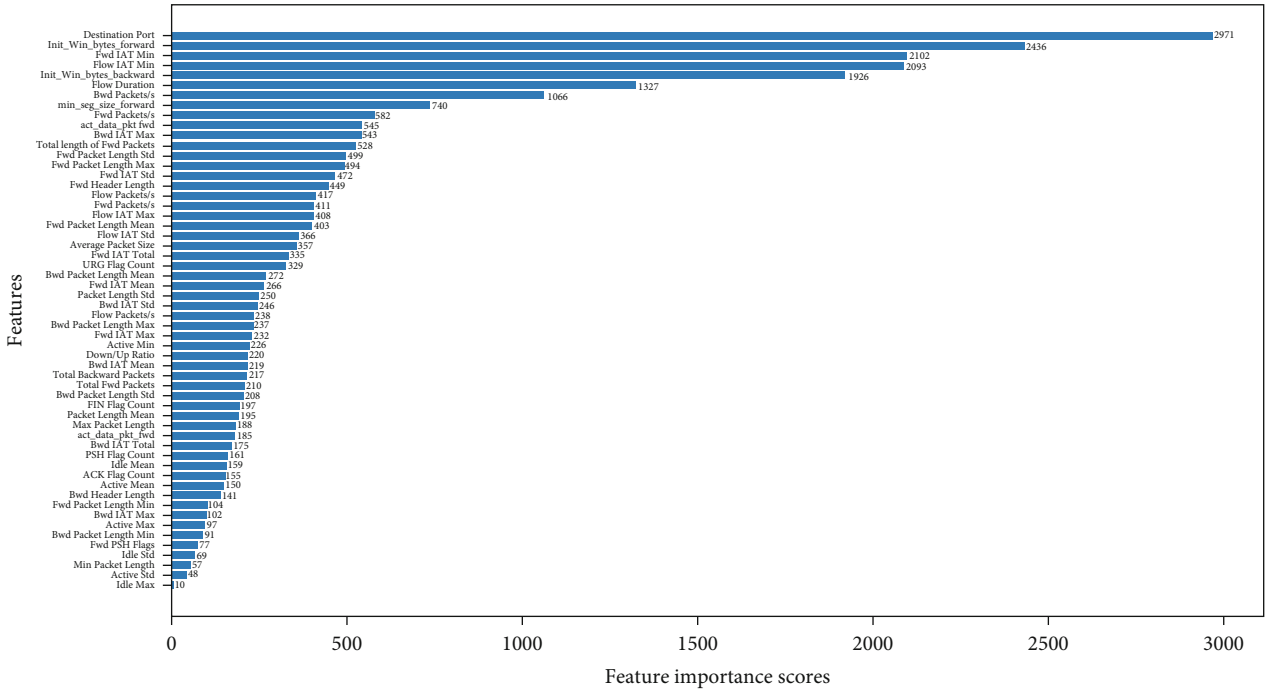

Figure 3: Rank diagram of different feature importance scores.

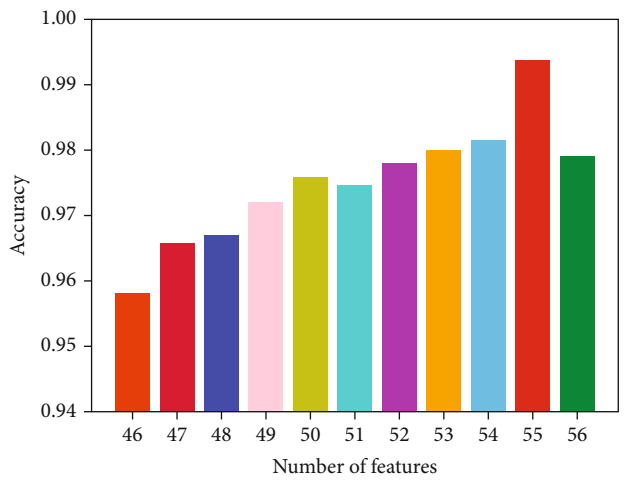

Figure 4: Classification results of different eigenvalues.

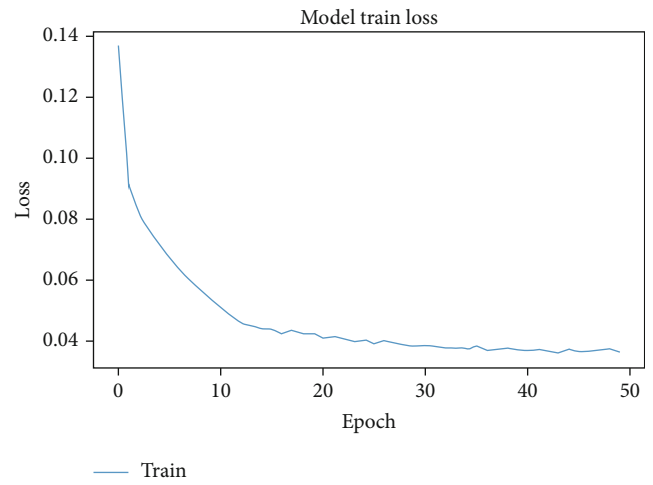

Figure 5: Loss results of different epochs.

an AlexNet style improved one-dimensional convolutional neural network [28] (I1DCNN), whose structure is illustrated in Figure 2. It primarily consists of four convolutional layers, two layers with maximum pooling, one layer with flattening, one softmax function output layer, and one full connection layer. To ensure the simultaneous reduction of crucial information, two convolution layers are used to fully extract features, followed by a maximum pooling layer for pooling processing. The previous convolution pooling operation is then carried out again. The output multidimensional data is then transformed into a one-dimensional array using the flattened layer. Use the softmax output function for output as you transition to the full connection layer.

The Adam optimization algorithm is used in this paper to optimize and create an improved one-dimensional convolutional neural network after output by the softmax output function. The Adam optimization algorithm is essentially a momentum method and RMSprop optimization algorithm. To ensure that all parameters are relatively stable, it dynamically modifies the learning rate of each parameter using the first-order moment estimation and second-order moment estimation of gradient [29]. Algorithm 2 illustrates the Adam optimization in this paper.

Utilizing the Adam optimization algorithm, the mean value of the gradient and the mean value of the gradient square are adaptively adjusted to improve the classification

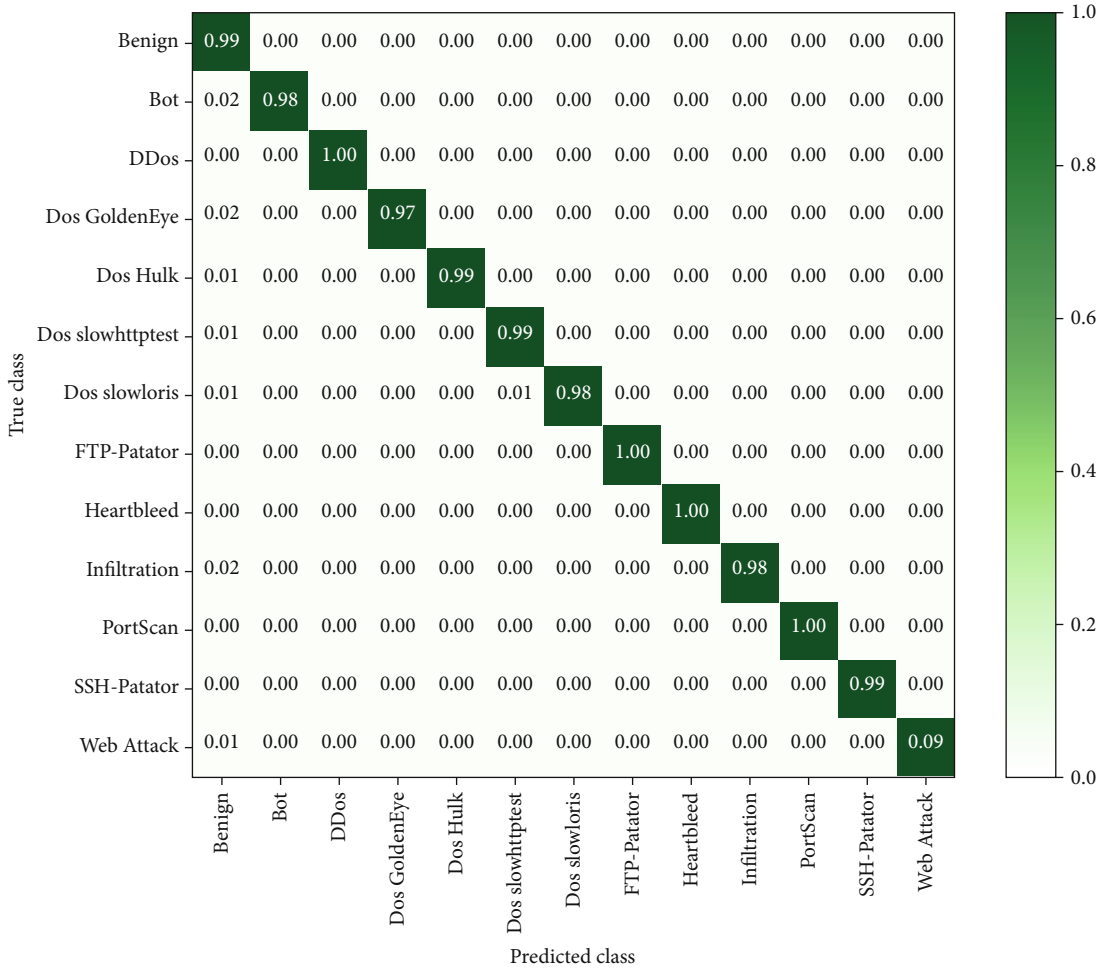

Figure 6: FS-I1DCNN results of model confusion matrix.

performance of the improved one-dimensional convolutional neural network created after feature screening.

4. Experimental Results and Analysis

4.1. Experimental Environment. The Windows Server 2016 operating system, Intel(R) Xeon(R) CPU E5-2650 v3@2.30 GHz, and the system type 64-bit operating system were the experimental environments used in this experiment. Python3.7.9 is the programming language and version. Pycharm and Anaconda are two third-party programs. Tenserflow-CPU is the primary deep learning framework, and Keras is used to build the network model. Machine learning libraries include sklearn and time.

4.2. Evaluation Indicators. In order to ensure the effectiveness of the experiment, accuracy, precision, recall, *F*1-score, and ROC curve were adopted as indicators to evaluate the performance of the machine learning model. Among them, true class (TP) refers to the number of positive cases correctly classified; false negative class (FN) refers to the number of positive cases misclassified as negative cases; false positive class (FP) represents the number of negative cases misclassified as positive cases; true negative class (TN) refers to the number of correctly classified negative cases; and the confusion matrix formed by it is shown in Table 2.

Accuracy, precision, recall, *F*1-score, and ROC curve were chosen as the indicators to assess the performance of the machine learning model in order to guarantee the efficacy of the experiment. False positive class (FP) represents the number of negative cases misclassified as positive cases. False negative class (FN) represents the number of positive cases incorrectly classified as negative cases, and true class (TP) refers to the number of positive cases correctly classified; the confusion matrix created by true negative class (TN), which is the quantity of correctly classified negative cases, is displayed in Table 2.

A better classifier has a higher accuracy, which primarily reflects the classifier's capacity to distinguish between positive and negative. The ability to distinguish between good and bad is improved with increased ability. The formula of accuracy is displayed in the following:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP}}. \tag{13}$$

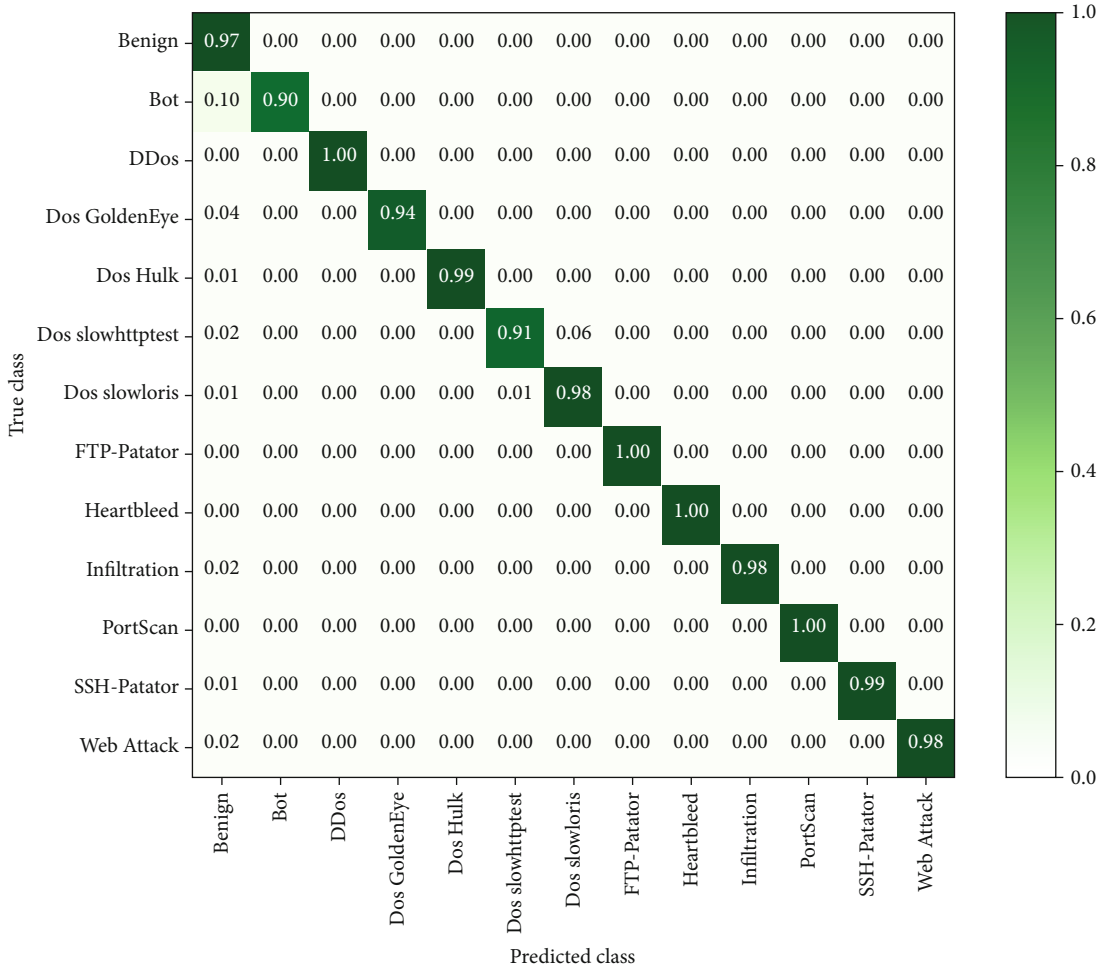

Figure 7: I1DCNN results of model confusion matrix.

Precision is the percentage of samples that contain only positive examples. Its formula is displayed in the following:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{14}$$

Recall is a measure of how many instances of a particular category were accurately classified into this category. Its formula is shown in the following:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{15}$$

The harmonic average of precision and recall is the $F1$ -score. The best model performance is represented by a value of 1, while the worst model performance is represented by a value of 0. In (16), its formula is displayed.

$$F1\text{-score} = \frac{2^{*}\text{precision}^{*}\text{recall}}{\text{precision} + \text{recall}}. \tag{16}$$

4.3. Comparison Experiment with Different Numbers of Features. The size of the number of features in the screening of the number of features has a significant impact on the outcomes of the experiment. A smaller number of screening may overlook the crucial features, while a larger number of screening may result in feature redundancy, which is of little significance to the experimental results. There are 56 features with varying degrees of contribution to the experiment, as shown in Figure 3's ranking of feature contribution scores for various features. Therefore, the preprocessed data in this paper are screened by the importance of features of 56 features by the size of contribution scores for relevant comparison experiments, and the results are shown in Figure 4. When the number of features is 55, it has the highest accuracy rate and has a clear advantage over other feature numbers. As the number of features decreases from 55, the accuracy rate gradually decreases. For this reason, 55 features were chosen as the final number of features for the experiment.

4.4. Model Results and Analysis. The prepreprocessed dataset was fed into the improved 1D convolutional neural network after feature importance selection. The experimental results are shown in Figure 5. When the number of iterations exceeds 40, the experimental results tend to be stable, so the epoch value is set to 40. The results of the confusion matrix obtained from the experiment are shown in Figure 6, which shows that the classification accuracy of each

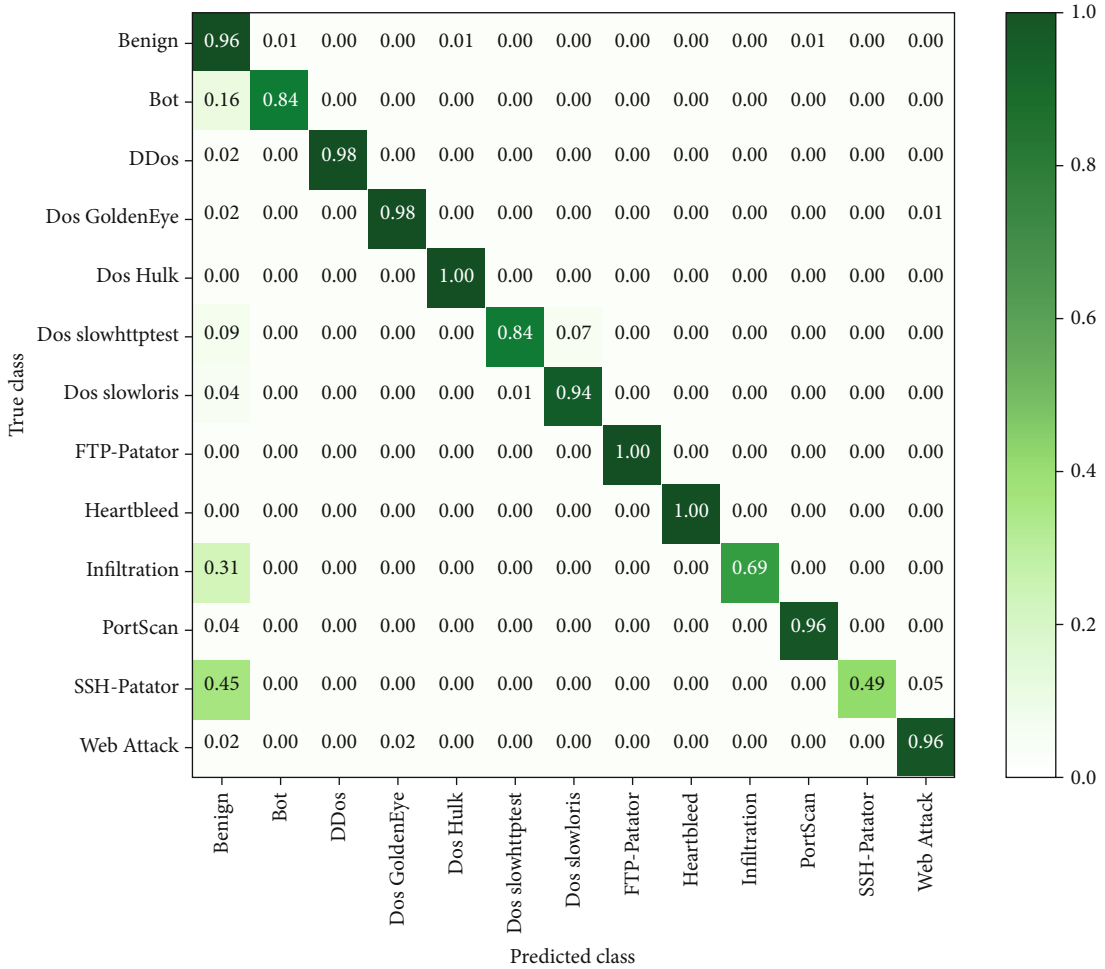

Figure 8: 1DCNN results of the model confusion matrix.

Table 3: Different algorithms correlation values.

Optimization algorithm	Accuracy	Precision	Recall	*F*1-score
Adatelta	0.9763	0.9701	0.9682	0.9698
Adagrad	0.9775	0.9721	0.9710	0.9733
Rmsprop	0.9823	0.9801	0.9636	0.9691
Nadam	0.9664	0.9558	0.9633	0.9627
Adam	0.9936	0.9887	0.9865	0.9877

category reaches more than 90%, including 100% for DDos, FTP-Patator, Heartbleed, and PortScan, and 98% for Dos Hulk, Dos slowloris, Infiltration, SSH-Patator, and Web Attack. The three types of malicious traffic also have accuracy of 90%, 94%, and 91%, respectively, for Bot, Dos GoldenEye, and Dos slowhttptest. In addition, the overall accuracy of the model also reaches 99.36%. So, it shows that the FS-I1DCNN intrusion detection model suggested in this paper has good performance in the classification results of each different class of malicious traffic and has good classification effect.

4.5. Comparison Experiment with I1DCNN and 1DCNN. This paper is experimentally compared with 1DCNN and I1DCNN

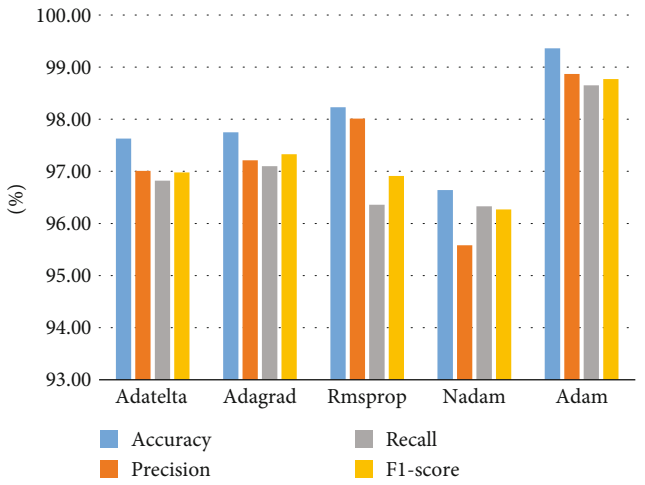

Figure 9: Comparison of different optimization algorithms.

to demonstrate the superiority of the FS-I1DCNN intrusion detection model. The results of the confusion matrix of the two are shown in Figures 7 and 8, with an overall accuracy rate of 96.42% and 98.69%, respectively. In conclusion, the

Table 4: Different classification model correlation values.

Classification model	Accuracy	Precision	Recall	*F*1-score
Rep+random forest[30]	0.9667	/	0.9448	/
PCA+SVM[31]	0.9291	/	0.9632	/
SVM+DBN[32]	0.97	0.98	0.97	0.97
PCA+LCNN[15]	0.9643	0.9589	0.9642	0.9606
ICNN[18]	0.98	0.98	0.98	0.98
FS-I1DCNN	0.9936	0.9887	0.9865	0.9877

I1DCNN model outperforms the 1DCNN model in terms of classification accuracy for all categories aside from benign and Dos Hulk, which are comparable to the 1DCNN model. It can be seen that the FS-I1DCNN model proposed in this paper has more or less improvement for each category when compared to the I1DCNN model without XGboost feature selection, and for a few categories, such as Bot and Dos slowhttptest, it has about 8% improvement. The I1DCNN intrusion detection model that is suggested in this paper has a notable improvement in accuracy for a select few categories and a better classification effect overall. Additionally, the average training times per epoch FS-I1DCNN compared to I1DCNN is 903 seconds and 988 seconds for these two methods, which is a full reduction of 85 seconds. This data demonstrates how the FS-I1DCNN intrusion detection model enhances classification effects while reducing dimensionality and operating times, further enhancing detection efficiency.

4.6. Comparative Experiment of Different Optimization Algorithms. When selecting the optimization algorithm of the model, different optimization methods have different adaptive fields and advantages, and the experimental results are also different. Considering that Adam can dynamically adjust the learning rate of each parameter by utilizing the gradient, allowing for the realization of self-adaptive learning and the achievement of a better classification effect when using this optimization algorithm. This paper conducts comparative experiments on various optimization algorithms based on the FS-I1DCNN intrusion detection model, and the outcomes are displayed in Table 3 and Figure 9. As can be seen, the Adam optimization algorithm has the highest classification accuracy when learning at the same rate, and it outperforms Adatelta in terms of accuracy rate, recall rate, and *F*1 score. The Adam optimization algorithm has the best classification effect because it is 1% to 3% higher than the Adagrad, Rmsprop, and Nadam optimization algorithms.

4.7. Comparison Experiment with Other Algorithms. Using the same dataset, it was compared cross-sectionally with other algorithmic models, such as random forest, SVM, DBN, LCNN, and ICNN, to determine how effective the FS-I1DCNN intrusion detection model proposed in this paper was. The results are shown in Table 4 and Figure 10. It is evident that the FS-I1DCNN intrusion detection model proposed in this paper has improved overall accuracy, precision, recall, and *F*1-score by an average of 4.36%, 1.57%, 2.21%, and 1.75% when compared to the other five models. All of the indexes are better than the other detection models, demonstrating the model's superior classification performance and applicability.

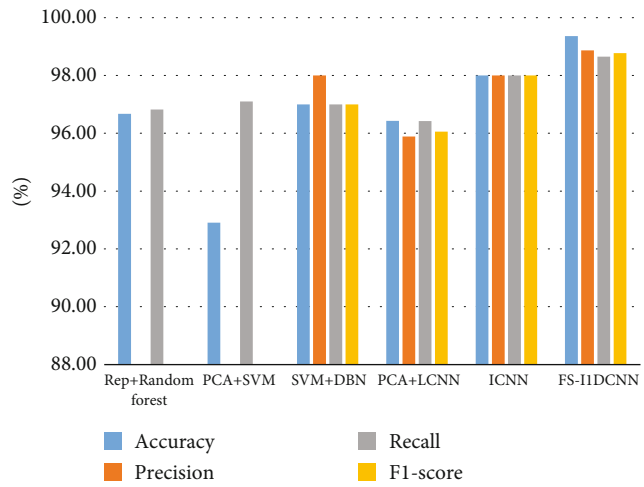

Figure 10: Comparison of the results of different models.

5. Conclusion

In order to address the problems with intrusion detection, a model based on FS-I1DCNN is proposed in this paper. After data processing, 55 features with a higher contribution are selected for the CIC-IDS-2017 dataset using the XGboost feature importance ranking method. These features are then fed into an improved one-dimensional convolutional neural network to finish the model's final classification task. The FS-I1DCNN intrusion detection model not only resolves the issue with the conventional machine learning approach of relying on expert experience to extract features but also enhances the detection efficiency of the model by using the XGboost feature screening method, reduces training time, and increases overall accuracy rate. The final experimental results demonstrate that the FS-I1DCNN intrusion model outperforms other intrusion detection models across all evaluation indices, achieving more than 97% classification accuracy in each category and 99.36% overall accuracy. In the future, the model will be used in actual network attack scenarios in order to further increase its detection effectiveness while maintaining high model classification performance.

Data Availability

The dataset used in this paper is open, which is proposed in reference.

Conflicts of Interest

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Acknowledgments

This research was funded by the New Infrastructure and University Informatization Research Project (Grant no. XJJ202205017).

References

[1] H. Hindy, R. Atkinson, C. Tachtatzis, J.-N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electronics*, vol. 9, no. 10, p. 1684, 2020.

[2] M. S. Benedetto, C. Anastasija, and A. N. Niels, "Training Guidance with KDD Cup 1999 and NSL-KDD Data Sets of ANIDINR: Anomaly- Based Network Intrusion Detection System," *Procedia Computer Science*, vol. 175, pp. 560–565, 2020.

[3] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artificial Intelligence Review*, vol. 55, no. 1, pp. 453–563, 2022.

[4] A. K. He, D. D. Kim, and M. R. Asghar, "Adversarial machine learning for network intrusion detection systems: a comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 538–566, 2023.

[5] A. Zeeshan, S. K. Adnan, W. S. Chean, A. Johari, and A. Farhan, "Network intrusion detection system: a systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, article e4150, 2021.

[6] H. Zhang, X. Y. Zhang, Z. Y. Zhang, and W. Li, "A review of intrusion detection models based on deep learning," *Computer Engineering and Applications*, vol. 58, pp. 17–28, 2021.

[7] X. Miao, Y. Liu, H. Zhao, and C. Li, "Distributed online one-class support vector machine for anomaly detection over networks," *IEEE Transactions on Cybernetics*, vol. 49, no. 4, pp. 1475–1488, 2019.

[8] Y. J. Chew, S. Y. Ooi, K. S. Wong, Y. H. Pang, and N. Lee, "Adoption of IP truncation in a privacy-based decision tree pruning design: a case study in network intrusion detection system," *Electronics*, vol. 11, no. 5, p. 805, 2022.

[9] A. V. S. Reddy, B. P. Reddy, L. Sujihelen, A. V. A. Mary, A. Jesudoss, and P. Jeyanthi, "Intrusion detection system in network using decision tree," in *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, pp. 1186–1190, Erode, India, 2022.

[10] H. W. Wang, G. Jie, and S. S. Wang, "An effective intrusion detection framework based on SVM with feature augmentation," *Knowledge-Based Systems*, vol. 136, pp. 130–139, 2017.

[11] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[12] P. Tao, Z. Sun, and Z. Sun, "An improved intrusion detection algorithm based on GA and SVM," *IEEE Access*, vol. 6, pp. 13624–13631, 2018.

[13] W. Wang, H. Dai, and S. Q. Dai, "Intrusion detection method based on feature optimization and BP neural network," *Computer Engineering and Design*, vol. 42, pp. 2755–2761, 2021.

[14] Y. D. Luo and L. Lu, "Network attack detection based on artificial neural network and genetic algorithm," *Computer Engineering and Design*, vol. 42, pp. 2446–2454, 2021.

[15] X. L. Zhang, G. Cheng, and W. C. Zhang, "Network traffic classification method based on improved deep convolutional neural network," *Scientia Sinica Informationis*, vol. 51, pp. 56–74, 2021.

[16] H. Yang and F. Wang, "Wireless network intrusion detection based on improved convolutional neural network," *IEEE Access*, vol. 7, pp. 64366–64374, 2019.

[17] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, p. 7986, 2022.

[18] M. X. Hang, W. Chen, and R. J. Zhang, "Abnormal traffic detection based on improved one-dimensional convolutional neural network," *Journal of Computer Applications*, vol. 41, pp. 433–440, 2021.

[19] T. Q. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, New York, NY, USA, 2016.

[20] A. Alsahaf, N. Petkov, V. Shenoy, and G. Azzopardi, "A framework for feature selection through boosting," *Expert Systems with Applications*, vol. 187, article 115895, 2022.

[21] A. Alsaleh and W. Binsaeedan, "The influence of salp swarm algorithm-based feature selection on network anomaly intrusion detection," *IEEE Access*, vol. 9, pp. 112466–112477, 2021.

[22] W. Cai and D. Hu, "QRS complex detection using novel deep learning neural networks," *IEEE Access*, vol. 8, pp. 97082–97089, 2020.

[23] Q. Zhu and X. Zu, "Fully convolutional neural network structure and its loss function for image classification," *IEEE Access*, vol. 10, pp. 35541–35549, 2022.

[24] Y. Wang, Y. Yang, W. Ding, and S. Li, "A residual-attention offline handwritten Chinese text recognition based on fully convolutional neural networks," *IEEE Access*, vol. 9, pp. 132301–132310, 2021.

[25] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISS*, vol. 1, pp. 108–116, 2018.

[26] J. C. Huang, G. Q. Zeng, G. G. Geng, J. Weng, K. D. Lu, and Y. Zhang, "Differential evolution-based convolutional neural networks: an automatic architecture design method for intrusion detection in industrial control systems," *Computers & Security*, vol. 132, article 103310, 2023.

[27] K. D. Lu, L. Zhou, and Z. G. Wu, "Representation-learning-based CNN for intelligent attack localization and recovery of cyber-physical power systems," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2023.

[28] N. Javaid, "A PLSTM, AlexNet and ESNN based ensemble learning model for detecting electricity theft in smart grids," *IEEE Access*, vol. 9, pp. 162935–162950, 2021.

[29] C.-H. Lin, Y.-C. Lin, and P.-W. Tang, "ADMM-ADAM: a new inverse imaging framework blending the advantages of convex optimization and deep learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–16, 2022.

[30] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, "A novel hierarchical intrusion detection system based on decision tree and rules-based models," in *2019 15th international conference on distributed computing in sensor systems (DCOSS)*, pp. 228–233, Santorini, Greece, 2019.

[31] H. Wang, Y. Xiao, and Y. Long, "Research of intrusion detection algorithm based on parallel SVM on spark," in *2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pp. 153–156, Macau, China, 2017.

[32] H. Zhang, Y. Li, Z. Lv, A. K. Sangaiah, and T. Huang, "A real-time and ubiquitous network attack detection based on deep belief network and support vector machine," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 3, pp. 790–799, 2020.