Journal of Computers Vol. 33 No. 4, August 2022, pp. 29-44
doi: 10.53106/199115992022083304003

Exploring Unsupervised Learning with Clustering and Deep Autoencoder to Detect DDoS Attack

Xuejun Zhang[1*], Jiyang Gai[1], Zhili Ma[2], Jinxiong Zhao[3], Hongzhong Ma[3], Fucun He[1], Tao Ju[1]

[1] School of Electronic and Information Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China
xuejunzhang@mail.lzjtu.cn
[2] State Grid Gansu Electric Power Company, Lanzhou 730000, China
[3] State Grid Gansu Electric Power Research Institute, Lanzhou 730070, China

Received 25 August 2021; Revised 28 December 2021; Accepted 14 February 2022

Abstract. With the proliferation of services available on the Internet, network attacks have become one of the serious issues. The distributed denial of service (DDoS) attack is such a devastating attack, which poses an enormous threat to network communication and applications and easily disrupts services. To defense against DDoS attacks effectively, this paper proposes a novel DDoS attack detection method that trains detection models in an unsupervised learning manner using preprocessed and unlabeled normal network traffic data, which can not only avoid the impact of unbalanced training data on the detection model performance but also detect unknown attacks. Specifically, the proposed method firstly uses Balanced Iterative Reducing and Clustering Using Hierarchies algorithm (BIRCH) to pre-cluster the normal network traffic data, and then explores autoencoder (AE) to build the detection model in an unsupervised manner based on the cluster subsets. In order to verify the performance of our method, we perform experiments on benchmark network intrusion detection datasets KDDCUP99 and UNSWNB15. The results show that, compared with the state-of-the-art DDoS detection models that used supervised learning and unsupervised learning, our proposed method achieves better performance in terms of detection accuracy rate and false positive rate (FPR).

Keywords: DDoS attack detection, autoencoder, clustering algorithm, unsupervised learning

1 Introduction

Recently, the Internet and communication terminals provide users with more convenient and efficient services, such as industrial control services, financial transactions, etc., for their works and life. However, more and more network attacks also occur frequently, which have posed a serious threat to network security. Distributed Denial of Services (DDoS) attack is one of the immense threats to disrupt communication networks and applications. It is reported that the total number of DDoS attacks exceeded 7.9 million in 2018 and grew to about 15.4 million by 2023 [1]. DDoS attack aims to continuously send a large number of traffic packets from multiple puppet machines to the target machine, resulting in the denial of normal services to users via exhausting the bandwidth resources or material resources of the victim machine, such as CPU or memory. Compared with other types of network attacks, DDoS attacks can hide their illegal traffic, making it a huge challenge to accurately detect such attack traffic.

Intrusion detection system is an effective strategy to detect DDoS attacks, which mainly uses firewalls to control normal access traffic. However, it is relatively single for detecting complicated DDoS attacks. Therefore, the detection technology for abnormal traffic is gradually introduced into DDoS attacks detection. As shown in Fig. 1, lots of safety devices recently use access control and abnormal traffic detection together to build the two-stage protection system. One of the most significant phenomena of a network attack is to generate a large amount of abnormal network traffic data, and the common detection method is mainly to perform statistical analysis on the features of the network layer such as the length of the network traffic, the average bit of the flow packet, the real-time rate of the port, and the throughput of the target machine. Wu et al. [2] proposed a method for detecting Low-rate DoS attacks by comparing and analyzing the pulse period, amplitude, and length of the network traffic sequence, which verifies that the analysis of network traffic features can achieve efficient detection of network attack traffic with different rates. Jing et al. [3] investigated the existing security-related data collection and analysis to detect network attacks. They firstly divided the relevant network security data into four categories. For each type of data, they provided a specific classification and discussed the pros and cons of each traffic feature in anomaly detection. And then the extracted traffic features were input into machine learning algorithms and neural network models to achieve traffic classification. These studies show that it is indispensable to analyze the patterns

* Corresponding Author

of traffic data to determine whether a network attack has occurred. This also makes the research on abnormal traffic detection methods become one of the most important research topics in recent years.

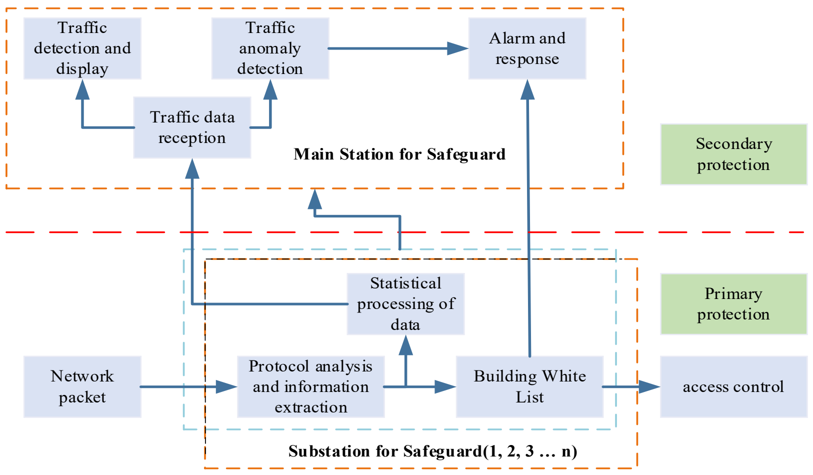

Traffic detection and display
Traffic anomaly detection
Alarm and response
Traffic data reception
Main Station for Safeguard
Secondary protection
Statistical processing of data
Primary protection
Network packet
Protocol analysis and information extraction
Building White List
access control
Substation for Safeguard(1, 2, 3 … n)

Fig. 1. Two-stage traffic access control and safety protection system

Machine learning-based methods [4] have recently made ideal achievements in anomaly detection. According to the datasets required for the training detection model, the existing anomaly detection methods that use machine learning can be divided into two categories: supervised learning-based and unsupervised learning-based. The methods based on supervised learning can achieve expected detection accuracy when being provided with well-labeled datasets. However, these detection methods require a large number of labeled samples that are difficult to obtain or high cost of acquisition. In addition, they cannot detect novel and unknown attacks. On the contrary, the methods based on unsupervised learning usually use data mining, correlation analysis, and pattern analysis to find the minimum subspace between normal data and abnormal data to identify whether the traffic is abnormal or not, which does not require labeled data for model training and can also detect unknown attacks. However, the downside of it is that these detection methods often require pure single-category training data to accurately learn the classification boundary between normal data and abnormal data. It suggests that the patterns of training data used for model training have a certain impact on the performance of the detection model. And detection model and threshold obtained by using traffic data with complex data patterns and features distribution could not accurately identify attack traffic and normal traffic. Therefore, high-quality training data is of great significance to unsupervised learning-based anomaly detection methods, and the inherent differences of the features of traffic data will inevitably affect the performance of the detection model.

Normal traffic generated by different software and programs corresponding to different network environments has different feature distribution and data patterns. However, the existing methods did not consider the impact of the different complex data patterns and features distribution of training data on the detection performance of the method based on unsupervised learning. Different from the state-of-the-art unsupervised learning-based attack detection methods, we proposed an improved detection method based on unsupervised learning to avoid the problems of using normal traffic data with unbalanced features distribution to the training detection model. In our proposed method, we firstly introduce a clustering algorithm to get the pure single-category data with similar features for the model training via pre-clustering the normal network traffic captured from the network environment. The object of the pre-clustering of the captured traffic data is to ensure the quality of the training data that is input into the unsupervised machine learning model, which assists in obtaining a more suitable detection threshold. In addition, we design an effective deep autoencoder (AE) neural network by adjusting the model structure and parameters. The main function of the deep AE neural network model is to learn the important features of cluster subsets so that the original input can be reconstructed to obtain the model output. Then, we obtain the detection threshold by calculating the difference between input and output. Through the threshold, the normal traffic and abnormal traffic can be distinguished accurately.

The essence of our method is to employ deep AE to search outliers by determining the difference between the input and output of the detection model (defined as reconstruction error in this paper). Our proposed method mainly includes the model training phase and the anomaly detection phase. In the model training phase, we use the BIRCH algorithm to pre-cluster the captured normal traffic data in an unsupervised way and obtain the clustering subsets with different patterns. The resulted clustering subsets are respectively input into the deep AE for reconstructing the input data through the process of “encoding-decoding” to obtain the average reconstruction error. Then the obtained reconstruction error is used as the detection threshold. In the anomaly detection phase, the test data was fed to the trained model for obtaining the reconstructed output of the input data. Furthermore, we use the mean squared error algorithm to calculate the reconstruction error between the input and its reconstruction output. When the reconstruction error is higher than the preset threshold, the traffic data is determined to be abnormal traffic. Otherwise, it is regarded as normal traffic.

The main contributions of our work are summarized as follows:

1) We select the appropriate clustering algorithm BIRCH to pre-cluster the captured normal traffic data in a way of unsupervised learning and use the cluster subsets with similar patterns as training data. The pre-clustering of the captured traffic data ensures the quality of the training data that is input into the unsupervised learning model, which assists in obtaining a more suitable detection threshold. Experiments show that the detection performance of the model trained with pre-clustered data outperforms the models trained on unprocessed data.

2) To build an abnormal traffic detection model with high detection accuracy and strong generalization ability, we design an effective deep AE neural network with five layers by adjusting the structure and parameters of the model. Based on the clustering subsets obtained by two benchmark network intrusion detection datasets KDDCUP99 and UNSWNB15, we trained detection models using basic AE, denoising AE and the AE model designed respectively. The experimental results show that the anomaly detection model trained with our AE model achieved better performance in terms of detection accuracy and FPR.

1.1 Outline

The rest of our paper is organized as follows. Section 2 introduces the existing machine learning-based methods for DDoS attacks detection. Then, we introduce the structure of the architecture of the proposed abnormal traffic detection model and the corresponding algorithms in section 3. Subsequently, the experimental results and analysis are presented in section 4. In section 5, we conclude the paper and discuss future work.

2 Related Work

In this section, we review some existing DDoS attack detection methods based on machine learning. We classify the DDoS detection methods into supervised learning-based methods and unsupervised learning-based methods systematically and discuss them separately.

2.1 Abnormal Traffic Detection Methods Using Supervised Learning

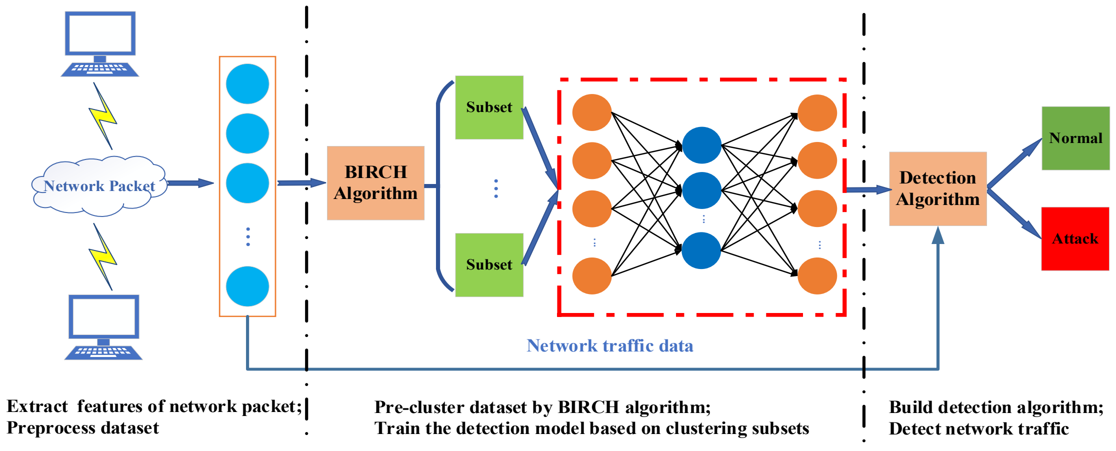
The abnormal traffic detection methods based on supervised learning are the classification task of data that use supervised machine learning algorithms. Saeed et al. [5] used a particle swarm optimization algorithm to select optimal features of traffic data and used a decision tree (DT) classification algorithm to implement a DDoS attack detection model. Fan et al. [1] proposed a DDoS attack detection model RF-SVM-IL based on machine learning. They used Random Forest (RF) and SVM to classify traffic data twice and utilized an incremental learning algorithm to filter the increased input samples and reduce the amount of data processed by the model. The model can effectively filter traffic samples that are easy to be misclassified when faced with massive attack traffic. Doshi et al. [6] built detection models based on K-Nearest Neighbors (KNN), RF, and DT to detect abnormal network traffic in IoT environments. Aamir et al. [7] firstly used an unsupervised clustering algorithm to label the collected traffic dataset, and then used supervised learning algorithms KNN, SVM, and RF to classify DDoS attacks in the network. Zekri et al. [8] proposed a classification method using DT based on the C4.5 algorithm to classify the DDoS attack traffic in cloud computing environments. Gumaei et al. [9] proposed a correlation-based feature selection (CFS) method to avoid irrelevant features traffic data, which used instance-based learning (IBL) algorithm to select optimal features and classify normal traffic and attack traffic based on supervised learning. Alhaidari et al. [10] used three kinds of machine learning algorithms (J48, Naive Bayes theorem, RF) to detect the attack traffic in Supervisory Control and Data Acquisition (SCADA) systems. Musumeci et al. [11] combined the capability of machine learning based on supervised learning and P4-enabled data planes to implement real-time DDoS attack detection, which achieved great performance regarding the accuracy and precision for all tested machines learning algorithms in most cases. Li et al. [12] proposed an intrusion detection system based on Online Sequence Extreme Learning Machine (OS-ELM) to detect anomaly traffic in Advanced Metering Infrastructure (AMI). Park et al. [13] used deep denoising autoencoder technology to achieve feature dimensional reduction and capture the nonlinear correlation between features, and the results show that the anomaly detection method based on autoencoder is superior to other traditional methods. Yuan et al. [14] proposed a DDoS attack detection method based on a cyclic deep neural network, which learned patterns from network traffic sequences and tracks network attack activities. Shaikh et al. [15] proposed a deep learning framework with autoencoder and recurrent neural network (RNN) methods for intrusion detection, which employed autoencoder to pre-classify the dataset and used long-term memory networks (LSTM) to perform the classification. The results proved that the method can effectively reduce false positives. Javaid et al. [16] proposed an intrusion detection method based on sparse autoencoders and soft-max regression, which used sparse autoencoders to extract features in an unsupervised way and took soft-max regression as a classifier to detect network traffic. However, these detection methods based on supervised learning require a large number of labeled samples for model training, which are difficult to obtain or high cost of acquisition. In addition, these detection methods do not guarantee the performance for detecting unknown attacks that do not appear in the training dataset.

2.2 Abnormal Traffic Detection Methods Using Unsupervised Learning

To address the defects of the supervised learning-based detecting models, the unsupervised learning-based network traffic detection methods were recently proposed to achieve anomaly detection by discovering the correlation between data features and data reconstruction through the mapping of data to a subspace. In the process of forming a subspace, the data with large reconstruction errors are identified as abnormal data. The classic Principal Component Analysis (PCA) method is a threshold-based anomaly detection algorithm. Novakov et al. [17] used hybrid PCA-Haar and wavelet algorithms to separate the high-dimensional space of network traffic data into non-intersecting subspaces corresponding to normal and abnormal traffic features and employed PCA and Haar Wavelet filtering to describe and analyze the data. Paffenroth et al. [18] used robust principal component analysis (RPCA) to detect anomalies, which achieved low FPR on individual packets. The results also further ssupported the hypothesis that the low dimensional subspace computed by RPCA is more representative of normal data. However, the majority of features of traffic data are nonlinear in the actual network environments, and the PCA algorithm is difficult to capture the nonlinear relationship between features due to its linear transformation process. Ali et al. [19] proposed a DDoS attack detection model that combined multi-layer autoencoders and multiple kernel learning (MKL) algorithm, which used nine deep autoencoders to build multiple kernel learning algorithms. Hence, the time overhead for model training and anomaly detection is high. Chen et al. [20] introduced a detection method based on unsupervised outlier detection, which achieved ideal detection accuracy. However, their method integrates multi-autoencoders to a single model to detect anomaly samples, thus leading to high time overhead for model training and anomaly detection. Yang et al. [21] proposed an AE-based DDoS attacks Detection Framework (AE-D3F) to learn the features of the training data and reconstruct the input of the model for obtaining a detection threshold. In AE-D3F, through the threshold, the normal traffic and abnormal traffic can be distinguished. However, the AE-D3F did not consider the impact of the different complex data patterns and features distribution of training data on the detection performance. The normal traffic generated by different software and programs corresponding to different network environments also has certain differences, such as the average length of the network traffic, the average bit of the flow packet, and the real-time rate of the port. Therefore, the detection model and threshold obtained by using an unsupervised learning-based method that does not consider the different complex data patterns and features distribution of training data is difficult to accurately identify attack traffic and normal traffic. Obviously, single-category and high-quality training data are of great significance to unsupervised learning-based anomaly detection methods. To solve this issue, we proposed to explore unsupervised learning with clustering and deep AE to accurately detect DDoS attacks. By leveraging the different data patterns and features distribution of training data, our proposed method firstly introduces a clustering algorithm to get the pure single-category data with similar features for the model training via pre-clustering the normal network traffic captured from the network environment. The pre-clustering of the captured traffic data ensures the quality of the training data that is input to the unsupervised learning model, which assists in obtaining a more suitable detection threshold. In addition, we design an effective deep autoencoder neural network to learn the important features of cluster subsets by adjusting the model structure and parameters, through which the original input can be reconstructed to obtain the model output. Then, we obtain the detection threshold by calculating the difference between the input and output, which can be employed to accurately distinguish between normal traffic and abnormal traffic.

3 Proposed Method

In this section, we present the architecture of our DDoS attack detection method as shown in Fig. 2.



Network Packet
BIRCH Algorithm
Subset
Subset
Detection Algorithm
Normal
Attack
Network traffic data
Extract features of network packet; Preprocess dataset
Pre-cluster dataset by BIRCH algorithm; Train the detection model based on clustering subsets
Build detection algorithm; Detect network traffic

Fig. 2. Architecture of the proposed detection model in this paper

As can be seen from Fig. 2, our proposed detection model mainly consists of traffic data preprocessing phase, detection model training phase, and an anomaly detection phase. In the traffic data preprocessing phase, the captured normal network traffic datasets first need to be normalized and standardized, and then pre-clustered into some clustering subsets with different patterns in an unsupervised way by using the BIRCH algorithm. In the detection model training phase, the resulting clustering subsets are respectively fed our deep autoencoder neural network for reconstructing the input data through the process of "encoding-decoding" to obtain the average reconstruction error. The final reconstruction error obtained is used as the detection threshold. In the anomaly detection phase, the test data is inputted into the trained model to get the reconstructed output of the input data. Then, the reconstruction error between the input and its reconstruction output was calculated by using the mean squared error algorithm. When the reconstruction error is higher than the preset threshold, the traffic data is determined to be abnormal traffic. Otherwise, it is determined to be normal traffic.

3.1 Data Pre-processing and Pre-clustering

In this paper, we use the open benchmark intrusion detection datasets KDDCUP99 and UNSWNB15 to perform experiments. Because the original dataset contains multiple types of data features, which directly affects the calculation of clustering features and the generation of clustering feature trees. Therefore, we firstly use Min-Max techniques [22] to normalize the numerical data in the dataset and transform the raw data linearly. Then, we map the normalized data to [0, 1]. The conversion function is expressed as:

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)} . \tag{1}$$

The principle of the abnormal traffic detection methods based on machine learning is to analyze the features of network traffic data. However, network traffic generated by different software and programs corresponding to different network environments also has different distribution features. Through analyzing the different patterns of normal traffic and abnormal traffic data, abnormal traffic is distinguished from normal traffic effectively. Thus, the inherent differences in the features of traffic data will inevitably affect the performance of the detection model. Our method firstly uses the BIRCH algorithm to pre-cluster the captured traffic data, through which the data with similar patterns will be clustered into clustering subsets. As a result, the clustering subsets are used as train data to train detection models, which can avoid the influence of unbalanced training data on the performance of the detection model.

The BIRCH algorithm [23] accomplishes high-quality clustering of large datasets with limited memory resources and is also more sensitive to abnormal data, which is conducive to the elimination of abnormal traffic data. In addition, the BIRCH algorithm uses cluster feature (CF) and CFTs to summarize a cluster and represent clustered hierarchies respectively. This enables clustering methods to operate on large databases with greater speed and scalability and also work well for incremental dynamic clustering.

CF and *CFT*s are the core of the BIRCH algorithm. A cluster feature consists of a triple group containing cluster information. Given a cluster that contains *N* *d*-dimensional traffic data $\{X_i\}$, in which $i = 1, 2, ..., N$, and *N* represents each feature of network traffic. The *CF* of the cluster is defined as $CF = (M, LS, SS)$, where *M* is the number of samples in a cluster, $LS = \sum_i^M X_i$ is the linear summation of *N* data and $SS = \sum_i^M X_i^2$ is the sum of the squares of *M* data. The additivity of CF is defined as follows: suppose that $CF_1 = (M_1, LS_1, SS_1)$ and $CF_2 = (M_2, LS_2, SS_2)$ are the different *CF* of two independent clusters, the additivity of two CFs can be represented as:

$$CF_1 + CF_2 = (M_1 + M_2, LS_1 + LS_2, SS_1 + SS_2) \ . \tag{2}$$

CFs can not only effectively reduce the storage space of data but also efficiently calculate all the indicators that form clustering decision in the BIRCH algorithm. For example, the distance between any two clusters can be expressed by CF as:

$$D = \sqrt{(2M * SS - 2LS^2) / M(M - 1)} \ . \tag{3}$$

By using D indicator, the BIRCH algorithm can achieve great clustering results. The phase of data preprocessing is presented in Algorithm 1.

Algorithm 1. Data preprocessing and pre-cluster

```
Input: normal traffic data $DS_{normal}$ and the cluster number range $N$
Output: clustering subsets $DS_{train}$
1: $X' \leftarrow \frac{X - \min(X)}{\max(X) - \min(X)}, \exists X \in DS_{normal}$   # Normalize the dataset using Min-Max technique
2: $temp\_score \leftarrow Infinity$   # $temp\_score$ is defined as initial value of Davies-Bouldin Index and is
                                         initialized to infinity
3: For $n$ in N do   # Calculate the optimal number of clusters
4:     $DB\_score \leftarrow Davies\_Bouldin\_score\ (X', n)$
5:   If $DB\_score < temp\_score$
6:       $temp\_score \leftarrow DB\_score$
7:       $n\_cluster \leftarrow n$
8:   End If
12: End For
13: $cluster\_subsets \leftarrow BIRCH(X', n\_cluster)$   # Cluster the data $X'$ using BIRCH algorithm
14: $DS_{train} \leftarrow cluster\_subsets$   # Cluster subsets are saved as training data
15: Return $DS_{train}$
```

In the algorithm 1, we use Min-Max technique to normalize and standardize the original dataset DS_{normal} and obtain the result X' (line 1). In the phase of data pre-cluster, we get the optimal number of clusters $n\_cluster$ (line 2 to line 12) by calculating the corresponding Davies-Bouldin score. Then, X' and $n\_cluster$ are inputted into the BIRCH algorithm as parameters to pre-classify the dataset (line 13 to line 14). Finally, we obtain clustering subsets with different sample sizes and label them with different IDs for subsequent model training.

3.2 Anomaly Detection Model Based on AE

AE is an unsupervised artificial neural network algorithm [24], which aims to extract the hierarchical features of high-dimensional input data to obtain high classification results. To effectively learn the data features of the clustering subsets obtained by the BIRCH algorithm and reconstruct the input data, we design a deep AE neural network model. Through the deep AE neural network model training, we obtain the minimized reconstruction error by which our method accurately distinguishes between normal traffic and malicious traffic. The framework of our deep AE neural network model is shown in Fig. 3.

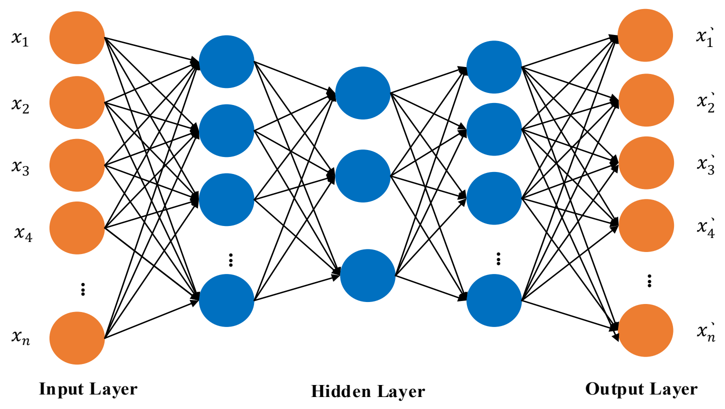


Fig. 3. The structure of the deep autoencoder neural network

As can be seen from the Fig. 3, our deep AE model consists of the encoding layer, hidden layer and decoding layer, which object is to find the minimized reconstruction error between input data $x = \{x_i\}_{i=1}^{n} \in \mathrm{R}^{n\times d}$ and its reconstructed data $x^{'} = \{x_i^{'}\}_{i=1}^{n} \in \mathrm{R}^{n\times d}$. To this end, we use MSE (mean square error) to calculate the reconstruction error θ:

$$\theta = L(x_i, x_i^{'}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_i^{'})^2 \quad . \tag{4}$$

The process of encoding is to reduce the dimension of a high-dimensional feature and compress a given input data into a specified dimension which is equal to the number of units of the hidden layer. The mapping function Z of the input data x to hidden layer is expressed as:

$$z_i = f(x) = s(\sum_{j=1}^{n} W_{ij}^{input} x_j + b_i^{input}) \quad . \tag{5}$$

Where x is the input vector, W is the weight matrix of the encoding layer, b is the bias matrix, and s is a non-linear activation function. In our method, we use Relu function as the activation function, which is expressed as $f(x) = \max(0, x)$. Correspondingly, the process of decoding is mainly to reconstruct the input data and decode the low-dimensional data of the hidden layer to the size of the original vector space. The mapping function of the decoding layer can be expressed:

$$x_i^{'} = g(z) = s(\sum_{j=1}^{n} W_{ij}^{hidden} z_j + b_i^{hidden}) \quad . \tag{6}$$

Where z is the output vector, W is the weight matrix of the hidden layer, b is the bias matrix, and s is a nonlinear activation function. The process of the deep AE model training is shown in Algorithm 2.

Algorithm 2. Training the detection model

```
Input: normal traffic dataset $DS_{train}$
Output: the detection model $AE\_model$ and the threshold $T$
1: $batch\_size \leftarrow 256$
2: $epochs \leftarrow 1$
3: $T \leftarrow 0$   # Threshold is initialized to 0.
4: While $epochs <= 10$ do
5:        $steps \leftarrow len(DS_{train}) / batch\_size$
6:        $loss \leftarrow 0$   # Loss is initialized to 0.
7:    For $step$ in range (0, $steps$) do
8:        $X \leftarrow DS_{train}[step * batch\_size, (step+1) * batch\_size]$   # Select the dataset that needs to be put into
              the model for each $step$ from $DS_{train}$.
9:        $En_{X\_batch} \leftarrow AE\_model.encoder(X)$   # Encoding of data.
10:       $X' \leftarrow AE\_model.decoder(En_{X\_batch})$   # Decoding of data.
11:       $l \leftarrow L(X, X')$ # Calculate the reconstruction error.
12:       $loss \leftarrow loss + l$
13:   End For
14:   $T \leftarrow loss / steps$   # Set detection threshold.
15:   $epochs \leftarrow epochs + 1$
16: End While
17: Return $AE\_model$, $T$
```

In algorithm 2, the $batch\_size$, $epochs$ and threshold T are initialized (line 1 to line 3). Then the clustering result DS_{train} is inputted into the proposed AE model for model training. Finally, the minimized average reconstruction error of train data can be obtained by the process of "encoding-decoding" (line 4 to line 16). Once the AE_model is trained, we save the trained model and determine the obtained reconstruction error of the train data as detection threshold T which is a minimized mean reconstruction error based on the mean square error function (line 14). Then we use test samples to evaluate the performance of the anomaly detection model. The detection phase is shown in Algorithm 3.

Algorithm 3. Testing the detection model

```
Input: test data $DS_{test}$, trained model $AE\_model$ and threshold $T$
Output: detection result $Detection\_result$
1: While $X_{test}$ in $DS_{test}$ do
2:    $X_{test}' \leftarrow AE\_model(X_{test})$
3:    $\theta \leftarrow L(X_{test}', X_{test})$   # Calculate the reconstruction error.
4:    If $\theta > T$   # Detected as anomaly traffic.
5:        $abnormal\_traffic \leftarrow X_{test}$
6:        $Detection\_result = abnormal\_traffic$
7:    Else   # detected as normal traffic.
8:        $normal\_traffic \leftarrow X_{test}$
9:        $Detection\_result = normal\_traffic$
```

```
10:     End If
11: End while
12: Return $Detection\_result$
```

In Algorithm 3, we firstly input the test data DS_{test} into the trained model AE_model to get its output (line 1 to line 2) and calculate the reconstruction error θ between the input and the output based on the reconstruction error function $\theta = L(x_1, x_i^{'})$ (line 3). Then, we detect the test sample by comparing the θ with the predetermined detection threshold *T*. If the θ is higher than the threshold *T*, the test sample is detected as abnormal traffic data, otherwise, it is detected as normal traffic data (line 4 to line 10).

4 Experimental Analysis and Results

In this section, we evaluate the performance of our method by performing comprehensive experiments on the intrusion detection datasets KDDCUP99 and UNSWNB15. We first evaluate the detection models based on un-clustered datasets and the clustering subsets respectively and then evaluate the performance of each detection model. Subsequently, we use basic AE and denoising AE to train the detection models for evaluating the performance of detection models that use different autoencoders. The basic AE consists of a single hidden layer, which aims to learn a model by extracting the representative data features using one hidden layer to reconstruct the original input data under specified constraints. The idea of denoising AE [25] is to train an autoencoder model to produce a robust input feature representing by reconstructing the original input after partially destroying the original input. It makes the trained model have better generalization ability. The corrupting process is generally to randomly select and replace the features in the input data with zeros at a certain proportion, or to add a certain noise to the input data. In addition, we compared our method with the recently supervised learning-based DDoS detection methods including Random Forest (RF) [6-7], Naive-Bayes (NB) [10], C4.5 DT (C4.5 Decision-Tree) [8], and RF-SVM-IL [1], and unsupervised learning-based DDoS detection methods including robust principal component analysis (RPCA) [18] and DDoS attacks Detection Framework (AE-D3F) [21].

4.1 Introduction of the Datasets

The KDDCUP99 dataset [26] from Columbia University's IDS Labs, contains 488,734 training data for 23 different types of attacks. Each sample has 38 features including source IP, source port, destination IP, destination port, transaction protocol, status, duration, and attack category. We select 92256 samples labeled with 'Normal' from the KDDCUP99 dataset as train data. To test the performance of the model, we select 10000 samples as test data which includes 5000 normal samples and 5000 DoS attack samples for testing the detection model.

The UNSWNB15 dataset [27] was collected by the Australian Cyber Security Centre (ACSC) using the IXIA Perfect-storm tool, which consists of 174,701 samples. Each sample has 49 features including source IP, source port, destination IP, destination port, transaction protocol, status, duration, and attack category. And its labels are classified into 'Normal' and 'DoS'. We select 93590 samples labeled with 'Normal' as train data, and 10000 samples as test data which includes 5000 normal samples and 5000 DoS attack samples.

4.2 The Experimental Results

The reasonable structure and hyper-parameter settings are indispensable for AE to obtain the good performance of the model. To this end, we perform comprehensive experiments to obtain an optimal hyper-parameter setting. The AE model of our method consists of a single input layer, a single output layer, and three hidden layers which correspond to n, $n/2$, $n/3$, $n/2$, n units, where *n* is the number of features of the train data. The batch size is set to 256, and the epoch is set to 10 (which is determined by experimental results). In addition, we use Relu as an activation function. The optimization algorithm is Adam with a 0.0001 learning rate. The loss function uses the mean square error (MSE) function. For basic AE, except that it contains only one hidden layer, the other parameter settings of the model are the same as our AE model. For denoising AE, the denoise parameter is set to 0.1, which means 10% of the input data is randomly replaced to zero. The other parameter settings of the denoising

AE model are the same as our AE model.

We compare the BIRCH clustering algorithm with the current mainstream clustering algorithms K-means and DBSCAN on the normal network traffic dataset (KDDCUP99). DBI (Davies-Bouldin Index) and CHI (Calinski-Harabasz Index) are used as the evaluation indicators of clustering performance. The DBI is used to evaluate the merits of clustering algorithms and also is known as a classification fitness indicator. Typically, the smaller the indicator, the better the clustering effect. CHI is mainly used to calculate the ratio of the dispersion between the clusters and the dispersion within the clusters, and it is an unsupervised evaluation indicator. Typically, the larger the CHI, the more dispersed and tighter the cluster, which means the better the clustering effect. Both indicators are respectively defined as equations (7) and (8):

$$DBI=\frac{1}{n}\sum_{i=1}^{n}\max_{i\neq j}\left(\frac{s_i+s_j}{d_{ij}}\right) \ , \tag{7}$$

$$CHI=\frac{tr(B_i)}{tr(W_i)}\times\frac{n-i}{i-1} \ . \tag{8}$$

Where s_i is the average distance from each point in cluster i to the cluster center, d_{ij} is the distance between the center of the cluster *i* and the center of the cluster *j*, *n* is the number of clusters, $tr(B_i)$ represents the trace of the deviation matrix between clusters, and $tr(W_i)$ represents the trace of the deviation matrix within clusters. The experiment results are shown in Table 1.

Table 1. Performance comparison of different clustering algorithms on DBI and CHI

Indicators	K-means	DBSCAN	BIRCH
DBI	0.9625	1.2094	0.8871
CHI	48666	23012	48266

As can be seen from Table 1, the BIRCH algorithm achieves the lowest DBI value which is followed by K-means and DBSCAN. For the CHI, the K-means gets the highest value 48666, BIRCH also gets 48266, and the DBSCAN only gets 23012. This is because it is difficult for DBSCAN to adjust the parameters of the global representation density, and when the amount of dataset increases, more memory support is required. Overall, we find that BIRCH and K-means have shown better performance. In the K-means algorithm, K is needed to be given in advance, and the selection of K is very difficult. Whereas, the BIRCH algorithm can determine the number of classification clusters after unsupervised learning. Therefore, we choose the BIRCH algorithm to pre-cluster the normal network traffic data.

In order to get a better clustering subset for model training, the dataset needs to be pre-processed by Min-Max techniques firstly. Then, we determine the best number of clusters by using the DBI score. The results are shown in Fig. 4.

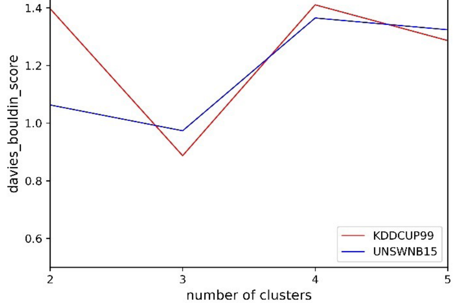


Fig. 4. Davies-Bouldin-Score

In Fig. 4, we adopt a different number of clusters to get different DBI scores on KDDCUP99 and UNSWNB15 datasets. It is observed that when the number of clusters is set to 3, the best clustering effect can be obtained. This is because the Davies-Bouldin indexes on both datasets are the smallest (0.9735 and 0.8871 respectively). Based on the optimal number of clusters, we use the BIRCH algorithm to pre-cluster the original normal traffic datasets. The results are shown in Table 2.

Table 2. Clustering results for the normal traffic data in KDDCUP99 and UNSWNB15

Datasets	Normal	ID 0	ID 1	ID 2
KDDCUP99	92256	69708	13096	9452
UNSWNB15	93590	40746	29070	23774

As can be seen from Table 2, for pre-clustering KDDCUP99, we categorize the original dataset from different data patterns and get three subsets of clusters with different sample sizes. Three subsets are labeled ID 0 (69708 samples), ID 1 (13096 samples), and ID 2 (9452 samples) respectively. For pre-clustering UNSWNB15, we also get three subsets of clusters with different sample sizes. Three subsets are labeled ID 0 (40746 samples), ID 1 (29070 samples), and ID 2 (23774 samples) respectively. All above six subsets will be used as train data to build the detection models respectively, and we select the model with optimal performance as our final detection model.

In our method, we choose the MSE of train data as the threshold to detect anomaly traffic. To verify the reason for our deep AE neural network model, we take the obtained six clustering subsets as train data to build detection models and use the test sample to calculate the distribution of reconstruction errors for normal traffic and abnormal traffic. Through the experiment, we predetermine the minimized training loss of the model as the detection threshold. In the detection phase, we input the test sample into the trained model to get its output and calculate the reconstruction error between the output and the original input of the model. The distributions of reconstruction errors of test samples are shown in Fig. 5, where the y-axis represents the reconstruction error corresponding to each traffic data, and the x-axis represents the type of traffic data.

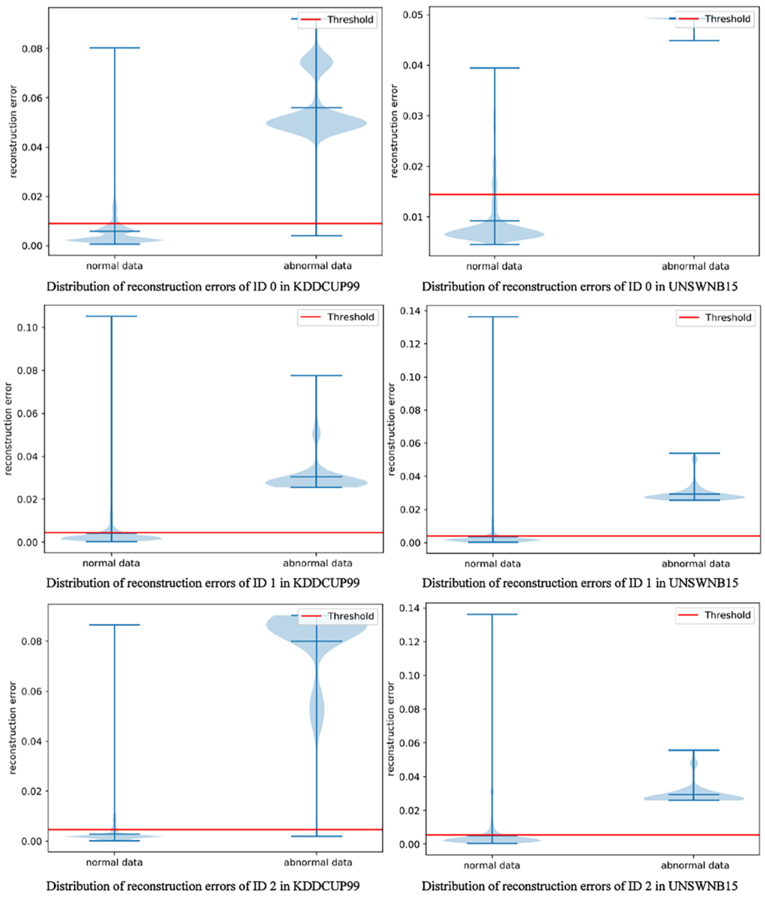


Fig. 5. Distribution of reconstruction errors for test samples

It is observed from Fig. 5 that the reconstruction error of abnormal traffic data is higher than that of normal traffic data. The reason for this phenomenon is that the test data that are similar to the training data will be reconstructed well, and thus have small reconstruction errors. On the contrary, the test data that are different from training data will have larger reconstruction errors because they are different from what the model was trained to reconstruct. So that the detection threshold obtained from the proposed model can effectively distinguish abnormal data and normal data. That means if the reconstruction error of the test data is higher than the predetermined detection threshold, the data is marked as abnormal traffic data. Otherwise, it is marked as normal traffic data.

To evaluate the overall performance of the detection model, Accuracy and FPR are used as evaluation indicators to compare the different methods. Both indicators can be obtained by the equations (9) and (10):

$$Accuracy = \frac{TP + FN}{TP + TN + FN + FP} \,, \tag{9}$$

$$FPR = \frac{TP}{TN + FP} \,. \tag{10}$$

In which TN (True Negative) is used for normal traffic to be detected as normal traffic. FN (False Negative) is used for abnormal traffic to be detected as normal traffic. TP (True Positive) is used for abnormal traffic to be identified as abnormal traffic. FP (False Positive) is used for normal traffic to be detected as abnormal traffic. We build the detection models using un-clustered data labeled with 'Normal' and clustering subsets (ID 0, ID1, and ID 2) based on KDDCUP99 and UNSWNB15. For each method, we take the average results of ten experimental as the final results. The experimental results are shown in Table 3.

Table 3. Performance of detection models based on different train datasets

Datasets	ID	Accuracy (%)	FPR (%)	Threshold
KDDCUP99	Normal	83.843	16.254	0.008636
	ID 0	92.938	8.966	0.241008
	ID 1	94.550	9.067	0.716162
	ID 2	99.157	0.373	1.173011
UNSWNB15	Normal	93.510	6.884	0.376727
	ID 0	96.700	3.541	0.419427
	ID 1	92.969	6.624	0.754252
	ID 2	93.014	6.483	0.349527

As can be seen in Table 3, for the KDDCUP99, the performance of the detection model trained with ID 2 is more efficient. The detection Accuracy and FPR are 99.157% and 0.373% respectively, and the corresponding values of the model based on un-clustered traffic data (Normal) are 83.843% and 16.254%. Especially, the detection accuracy and FPR of the model based on clustering subset ID 2 are 15.304% higher and 15.881% lower respectively than that of the model trained with the un-clustered data. Therefore, it is obvious that the detection threshold obtained based on ID 2 achieves better detection performance, which is selected as the final threshold to detect anomaly traffic. On UNSWNB15, the performance of the detection model trained with ID 0 is more efficient, the detection Accuracy and FPR are 96.700% and 3.541% respectively. Therefore, we use the threshold obtained based on ID 0 as the final threshold to detect anomaly traffic. Through the experimental results, we observed that the detection models trained with clustering subsets perform greater performance than the detection models trained with un-clustered data. This is because the detection model trained on pure single-category data with similar features is more sensitive to abnormal data. If the test data is a little different from training data, the detection model cannot reconstruct the test data well, thus leading to a larger reconstruction error. As a result, the obtained detection threshold cannot effectively distinguish between abnormal data and normal data. Besides, from the distribution of the detection threshold of all experimental results, we observed that the model with a larger threshold achieves better detection performance. Because the threshold is too small, the model will misjudge normal traffic as abnormal traffic. On the contrary, if the threshold is too large, the model will misjudge abnormal traffic as normal traffic. This means it is critical to determine a suitable threshold for detecting anomaly traffic.

A Performance of Detection Models Trained with Different AEs

To evaluate the performance of the detection model based on our deep AE model, we compare our deep AE model with basic AE and denoising AE on the six clustering subsets. The experimental results are the average of the ten experimental results corresponding to the six clustering subsets, shown in Table 4.

Table 4. Performance of detection models trained with different AE on KDDCUP99 and UNSWNB15 datasets

Datasets	Subsets	Models	Accuracy (%)	FPR (%)	Threshold
KDDCUP99	ID 0	Our method	**92.938**	**8.966**	0.241008
		Basic AE	87.403	14.095	0.004609
		Denoising AE	88.409	11.741	0.005672
	ID 1	Our method	**94.550**	**9.067**	0.716162
		Basic AE	86.546	14.744	0.015498
		Denoising AE	76.807	23.659	0.003953
	ID 2	Our method	**99.157**	**0.373**	1.173011
		Basic AE	93.539	10.850	0.017281
		Denoising AE	77.309	22.767	0.003953
UNSWNB15	ID 0	Our method	**96.700**	**3.541**	0.419427
		Basic AE	88.299	11.776	0.019681
		Denoising AE	82.562	17.514	0.003322
	ID 1	Our method	**92.969**	**6.624**	0.754252
		Basic AE	85.172	14.924	0.023540
		Denoising AE	79.630	20.503	0.010530
	ID 2	Our method	**93.014**	**6.483**	0.349527
		Basic AE	84.715	15.374	0.025460
		Denoising AE	81.894	18.224	0.012008

As can be seen from Table 4, our deep AE model achieves the best Accuracy and FPR of three comparison AE models on both datasets. For example, on KDDCUP99, the detection accuracy of our deep AE model trained using ID 0 achieves 92.938%, and the corresponding FPR achieves 8.966%. On UNSWNB15, the detection accuracy of our deep AE model trained using ID 0 achieves 96.700%, and the corresponding FPR achieves 3.541%. The reason for this phenomenon is that the higher model complexity of our deep AE model can accurately capture more heterogeneous patterns, instead of simply averaging between normal and abnormal patterns. Besides, the detection threshold of our AE models is slightly higher than that obtained by other comparison AE models on both datasets, which indicates our method achieves more appropriate reconstruction for normal traffic data, and the detection threshold obtained by the deep AE model is more appropriate for detecting anomaly traffic. In addition, it can be seen that the basic AE model achieves better Accuracy and FPR than the denoising AE model. For example, on KDDCUP99, the detection accuracy of the basic AE model trained using ID 1 achieves 86.546% which is 9.739% higher than that obtained by denoising AE model, and the corresponding FPR achieves 14.744% which is 8.915% lower than that obtained by denoising AE model. On UNSWNB15, the detection accuracy of the basic AE model trained using ID 1 achieves 85.172% which is 5.542% higher than that obtained by denoising AE model, and the corresponding FPR achieves 14.924% which is 5.579% lower than that obtained by denoising AE model. However, on ID 0 of KDDCUP99, the denoising AE model achieves slightly better Accuracy and FPR than the basic AE model, the reason for this exception might be the differences in the features that determine the data category in ID 0 of KDDCUP99 are more obvious than other datasets. Therefore, the operation of adding noise to the training data not only did not affect the detection accuracy of the model but instead improved the robustness of the model, resulting in a slight increase in the detection accuracy.

B Performance Comparison and Analysis

Table 5 shows the comparison of the average accuracy and FPR of our proposed method with two recent unsupervised learning-based DDoS attack detection methods RPCA [18] and AE-D3F [21]. We train RPCA and AE-D3F models by using normal samples of the KDDCUP99 and UNSWNB15, respectively.

Table 5. The experimental results compared with the methods based on unsupervised learning

Datasets	Methods	Accuracy (%)	FPR (%)
KDDCUP99	Our method	**99.157**	**0.373**
	RPCA	96.436	3.126
	AE-D3F	94.475	4.728
UNSWNB15	Our method	**96.700**	**3.541**
	RPCA	92.542	9.490
	AE-D3F	92.339	7.094

From Table 5, we can see that our method is better than both RPCA and AE-D3F models in terms of the Accuracy and PPR on two datasets. On the KDDCU99 dataset, our method achieves 99.157% accuracy and 0.373% FPR. The accuracy of our method is 2.822% and 4.956% higher than that of RPCA and AE-D3F, the FPR of our method is 5.949% and 3.553% lower than that of RPCA and AE-D3F, respectively. Similarly, on the UNSWNB15 dataset, the accuracy of our method is 4.493% and 4.723% higher than that of RPCA and AE-D3F, the FPR is 62.687% and 50.099% lower than that of RPCA and AE-D3F respectively.

This is probably because we use the BIRCH cluster algorithm to pre-classify network traffic data with similar distribution features captured from the network environment. The pre-clustering of the captured traffic data ensures the quality of the training data that is input into the unsupervised learning model. Based on the clustering subsets, we design a deep autoencoder to train the detection model, which can learn the high-dimensional features of the pre-clustered subsets and assist in obtaining a more suitable detection threshold. Thus, we achieve better performance in terms of detection accuracy and FPR.

In Table 6, we compare the proposed method with the recently supervised learning-based DDoS attack detection methods (NB [10], RF [6-7], C4.5_DT [8], and RF-SVM-IL [1]). Because the mentioned methods need labeled data samples to train the detection model. Therefore, we use KDDCUP99 and UNSWNB15 datasets with labels to train NB, RF, C4.5_DT, and RF-SVM-IL models, respectively. For the method proposed in this paper, we use the experimental results obtained based on ID 2 of KDDCUP99 and ID 0 of UNSWNB15 as the comparison.

Table 6. The experimental results compared with the methods based on supervised learning

Datasets	Indicators	NB	RF	C4.5_DT	RF-SVM-IL	Our method
KDDCUP99	Accuracy (%)	93.188	94.647	97.101	97.471	**99.157**
	FPR (%)	9.648	7.118	3.886	2.774	**0.373**
UNSWNB15	Accuracy (%)	84.945	92.482	93.621	94.852	**96.700**
	FPR (%)	14.510	7.034	6.445	5.038	**3.541**

From Table 6, we can see that our method achieves the optimal performance among all the comparison methods with regard to the accuracy and FPR on both datasets. For example, on the KDDCUP99 dataset, the detection accuracy of our method is the highest, at 99.157%. Correspondingly, the accuracy of NB, RF, C4.5_DT, and RF-SVM-IL are 93.188%, 94.647%, 97.101%, and 97.471%, respectively. And the FPR of our method achieves 0.373% which is lower than that of NB, RF, C4.5_DT, and RF-SVM-IL. Similarly, on the UNSWNB15 dataset, our method achieves the best detection accuracy of 96.700%. Correspondingly, the accuracy of NB, RF, C4.5_DT, and RF-SVM-IL are 84.945%, 92.482%, 93.621%, and 94.852%, respectively. And the FPR of our method achieves 3.541% which is lower than that of LR, NB, C4.5_DT, and RF-SVM-IL.

We can conclude that our method is more effective than other methods. Meanwhile, the RF-SVM-IL and C4.5_DT achieve greater performance than NB, RF. Tt is because RF-SVM-IL used Random Forest and SVM to classify traffic data twice and effectively filter traffic samples that are easy to be misclassified. Similarly, C4.5_

DT improved the decision algorithm in the feature selection stage. Therefore, the optimal features can be selected to divide the decision boundary to obtain better classification results, and it also proves that the preprocessing of the training data directly affects the detection performance of the model. Indeed, the methods based on supervised learning are essentially a kind of classification method which need large amounts of high-quality data training data with labels for building detection model. Especially, these methods can achieve expected detection accuracy when being provided with well-labeled datasets. However, our method uses deep AE to learn the high-dimensional features of the pre-clustered subsets through the processes of "encoding" and "decoding", and uses an appropriate threshold for detecting anomaly traffic. As a result, our detection model reduces the probability of classifying normal traffic as abnormal traffic and guarantees higher detection accuracy and lower FPR. Besides, our method is based on unsupervised learning, which does not need to label the training samples. Thus, our method not only saves a lot of costs for labeling data but also can detect unknown attacks.

5 Conclusion

This paper design an improved DDoS attack detection model based on unsupervised learning, which aims to achieve more accurate and efficient DDoS attack detection in the network communication system. To this end, our method firstly uses the BIRCH algorithm to cluster network traffic data with similar features and utilizes a greater cluster subset as training data. Then, we train the detection model using deep AE in an unsupervised way and use the minimized reconstruction error of training data as a threshold to detect abnormal traffic. We compare our method with recent DDoS detection methods based on supervised learning and unsupervised learning, and the experiments on KDDCUP99 and UNSWNB15 show that our method is superior to the comparison methods in terms of accuracy and FPR. In addition, our method provides a practical guideline for developing network intrusion detection systems based on autoencoder and significantly contributes to the exploration of unsupervised learning techniques for various network intrusion detection systems.

The method proposed in this paper also has limitations and needs to be further improved to adapt to a more extensive and complex network environment: (1) our method cannot detect the specific type of network attack. Next, We intend to design an abnormal traffic detection method based on semi-supervised learning that can both detect specific types of attacks as well as unknown attacks; (2) to improve the generalization capability of the model, we will consider adjusting our detection model according to different network traffic with various rates in different environments; and (3) to achieve the usability of the method, we are going to build a real-time and adaptive network attack detection software based on the algorithm proposed in this paper.

6 Acknowledgement

This work was partly supported by the NSFC project (grant no. 61762058 and no. 61862037), A Hundred Youth Talents Training Program of Lanzhou Jiaotong University, the Science and Technology Project of Gansu Province (grant no. 21JR7RA282, and no. 20JR5RA404), and the Science and Technology Project of State Grid Gansu Electric Power Research Institute (grant no. 52272219100P).

References

[1] J.-Y. Fan, G.-Q. Yang, J.-Y. Gai, DDoS attack detection system based on RF-SVM-IL model under SDN, Journal of Computers 32(5)(2021) 31-43.

[2] Z. Wu, Q. Pan, M. Yue, Sequence alignment detection of TCP-targeted synchronous low-rate DoS attacks, Computer networks 152(APR.7)(2019) 64-77.

[3] X. Jing, Z. Yan, W. Pedrycz, Security Data Collection and Data Analytics in the Internet: A Survey, IEEE Communications Surveys & Tutorials 21(1)(2018) 586-618.

[4] N. Moustafa, J.-K. Hu, J. Slay, A holistic review of network anomaly detection systems: A comprehensive survey, Journal of Network and Computer Applications 128(2019) 33-55.

[5] A.-A. Saeed, N. Jameel, Intelligent feature selection using particle swarm optimization algorithm with a decision tree for DDoS attack detection, International Journal of Advances in Intelligent Informatics 7(1)(2021) 37-48.

[6] R. Doshi, N. Apthorpe, N. Feamster, Machine Learning DDoS Detection for Consumer Internet of Things Devices, in: Proc. of 2018 IEEE Security and Privacy Workshops, 2018.

[7] M. Aamir, S. Zaidi, Clustering based Semi-Supervised Machine Learning for DDoS Attack Classification, Journal of King Saud University Computer & Information Sciences 33(4)(2021) 436-446.

[8] M. Zekri, S. Kafhali, N. Aboutabit, Y. Saadi, DDoS attack detection using machine learning techniques in cloud computing

environments, in: Proc. of 3rd International Conference of Cloud Computing Technologies and Applications, 2017.
[9] A. Gumaei, M.-M. Hassanand, S. Huda, A robust cyberattack detection approach using optimal features of SCADA power systems in smart grids, Applied Soft Computing 96(4)(2020) 106658:1-17.
[10]F.-A. Alhaidari, E.-M. AL-Dahasi, New approach to determine DDoS attack patterns on SCADA system using machine learning, in: Proc. of 2019 International Conference on Computer and Information Sciences (ICCIS), 2019.
[11]F. Musumeci, A.-C. Fidanci, F. Paolucci, Machine-Learning-Enabled DDoS Attacks Detection in P4 Programmable Networks, Journal of Network and Systems Management 30(1)(2021) 1-27.
[12]Y.-C. Li, R.-X. Qiu, S.-T. Jing, Intrusion detection system using Online Sequence Extreme Learning Machine (OS-ELM) in advanced metering infrastructure of smart grid, PloS one 13(2)(2018) 1-16.
[13]S. Park, S. Seo, J. Kim, Network Intrusion Detection Using Stacked Denoising Autoencoder, Advanced Science Letters (2017) 9907-9911.
[14]X.-Y. Yuan, C.-H. Li, X.-L. Li, DeepDefense: identifying DDoS attack via deep learning, in: Proc. of IEEE International Conference on Smart Computing (SMARTCOMP), 2017.
[15]R.-A. Shaikh, S.-V. Shashikala, An Autoencoder and LSTM based Intrusion Detection approach against Denial of Service attacks, in: Proc. of 1st International Conference on Advances in Information Technology, 2019.
[16]A. Javaid, Q. Niyaz, W. Sun, M. Alam, A Deep Learning Approach for Network Intrusion Detection System, in: Proc. of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies, 2016.
[17]S. Novakov, C-H. Lung, I. Lambadaris, Studies in applying PCA and wavelet algorithms for network traffic anomaly detection, in: Proc. of 2013 IEEE 14th International Conference on High Performance Switching and Routing (HPSR), 2013.
[18]R. Paffenroth, K. Kay, L. Servi, Robust PCA for anomaly detection in cyber networks. <https://arxiv.org/pdf/1801.01571.pdf>, 2018 (accessed 18.01.04).
[19]S. Ali, Y.-C. Li, Learning Multilevel Auto-encoders for DDoS Attack Detection in Smart Grid Network, IEEE Access 7(2019) 108647-108659.
[20]J.-H. Chen, S. Saket, A. Charu, T. Deepak, Outlier detection with autoencoder ensembles, in: Proc. of the 2017 SIAM International Conference on Data Mining, 2017.
[21]K. Yang, J. Zhang, Y. Xu, DDoS attacks detection with autoencoder, in: Proc. of NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, 2020.
[22]Z. Ihsan, M.-Y. Idris, A.-H. Abdullah, Attribute normalization techniques and performance of intrusion classifiers: A comparative analysis, Life Science Journal 10(4)(2013) 2568-2576.
[23]B. Lorbeer, A. Kosareva, B. Deva, Variations on the Clustering Algorithm BIRCH, Big Data Research 11(2018) 44-53.
[24]Y. Han, Y. Ma, J. Wang, Research on ensemble model of anomaly detection based on autoencoder, in: Proc. of 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS), 2020.
[25]W. Jin, B. Sun, Z. Li, S. Zhang, Z. Chen, Detecting Anomalies of Satellite Power Subsystem via Stage-Training Denoising Autoencoders, Sensors 19(14)(2019) 3216:1-13.
[26]M. Tavallaee, E. Bagheri, W. Lu, A detailed analysis of the KDD CUP 99 data set, in: Proc. of 2009 IEEE symposium on computational intelligence for security and defense applications, 2009.
[27]N. Moustafa, J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: IEEE Proc. of 2015 military communications and information systems conference (MilCIS), 2015.