IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 September 2022, accepted 23 September 2022, date of publication 26 September 2022, date of current version 4 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3210189

RESEARCH ARTICLE

Traffic Anomaly Detection in Wireless Sensor Networks Based on Principal Component Analysis and Deep Convolution Neural Network

CHENGPENG YAO, YU YANG, KUN YIN, AND JINWEI YANG
College of Information Engineering, Engineering University of PAP, Xi'an 710086, China

Corresponding author: Yu Yang (miaoyude@163.com)

This work was supported by the Fundamental Research Fund of the School of Information Engineering, Engineering University of PAP, under Grant WJY202130.

ABSTRACT With the popularity of wireless networks, wireless sensor networks (WSNs) have advanced rapidly, and their flexibility and ease of deployment have resulted in more security concerns, making it critical to research network intrusion prevention for WSNs. Denial of service (DoS) is a common network attack, achieving its goal by bringing down the target network. A DoS attack on WSNs devices with limited resources would be fatal. This paper proposes a method based on principal component analysis (PCA) and a deep convolution neural network (DCNN) for DoS traffic anomaly detection in WSNs, based on the vulnerability of WSNs to attacks and the limited storage space of their devices. Compared with the conventional deep learning structure, the proposed model has a lightweight structure and more effective feature extraction capability, which can effectively detect network abnormal traffic in WSNs devices with limited storage capacity. To assure the effectiveness of the proposed model, receiver operating characteristic (ROC) curves, various classification metrics, and confusion matrices are used to verify the classification results of the model. Through experimental comparison, the proposed model, with small model size, outperforms other mainstream abnormal traffic detection models in terms of classification effect.

INDEX TERMS Wireless sensor networks, denial of service, network attack, principal component analysis, deep convolution neural network.

I. INTRODUCTION

Wireless sensor networks (WSNs) have evolved rapidly in recent years, and have become one of the critical areas for research in network applications [1]. Their low cost and ease of deployment have enabled them to be used in various smart sensor devices in our lives, such as in-vehicle communications, smart homes, and remote monitoring. With the rapid development and widespread application of smart sensor network devices, many researchers have focused on the security of WSNs [2]. WSNs can collect and deliver environmental information in real-time, and their flexible and efficient features make people's lives easier. But at the same time, it also has the disadvantages of limited power, low processing capabilities, security, and data trustworthiness [3], these shortcomings lead it can be hacked, corrupted, or exploited at any time [4], and once a network has been attacked, whether it is via personal devices or public resources, the loss is enormous. Thus, the detection of traffic anomalies in WSNs has become increasingly important.

Anomalous traffic in a network varies, and various network attacks exist. Denial of Service (DoS) attack is one of the most common and achievable attacks in practice due to its easy implementation [5], it exhausts the resources of the target system by generating large traffic and prevents the server from processing legitimate requests from normal users [6]. There are two main forms of DoS attacks. One is to create a large number of useless traffic, resulting in host network congestion; the other is the use of network protocol implementation flaws so that users can not receive

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

traffic information normally, such as the destruction of TCP protocol connection session during the three handshakes' process. When the DoS attack takes effect, the normal operation ability of the host's CPU is reduced, and the memory allocation is wrong, resulting in the depletion of network bandwidth, and even affecting the network systems such as routers and firewalls. In severe cases, there will be downtime, resulting in the host not running normally. The most common distributed DoS (DDoS) is to attack the service availability of single or multiple victim systems through collaborative attacks [7], and ultimately make the computer lose the ability of normal service. For example, in 2018, GitHub, an online code management platform, suffered a DDoS attack with a transmission rate of 126.9 million per second at the network's peak. Fortunately, GitHub had a DDoS defense system in place, preventing further damage [8]. DoS attacks, which are common and vicious, will be fatal in resource-limited WSNs devices, causing severe damage. Thus, their accurate and efficient detection in WSNs is critical.

Currently, network environments are complex, network attacks are diverse, and multivector attacks using multiple protocol combinations for DoS attacks are becoming more common [9], which all add to the difficulty of network anomaly traffic detection. Network traffic anomaly detection using machine learning (ML) and deep learning (DL) has been proven to be reliable. ML has developed into an effective method for processing complex data in the past few years [10], and it is widely used in anomaly traffic detection because of its simplicity and efficiency. For instance, Monshizadeh *et al.* [11] used a combination of conditional variational autoencoder (CVAE) and random forest (RF) for network traffic anomaly detection, where CVAE learns the similarity between input features, and then RF is used to classify the anomalous traffic. Ma *et al.* [12] proposed a kernel support vector machine (SVM)-based network traffic anomaly detection method, as well as optimized model hyperparameters, to classify anomalous traffic. Iranmanesh *et al.* [13] used the time-homogeneous semi-Markov process to predict the likelihood of the accuracy of vehicle mobility patterns in the malicious detection of traffic flow and then calculated the weight factor through Cloudlets to determine whether it was malicious traffic flow. However, ML methods are inadequate for feature learning, especially in today's more complex and variable network environments. As an important tool for data mining and reconstruction [14], DL methods can extract high-level network traffic features, so the use of DL methods for traffic anomaly detection has become mainstream nowadays. For instance, Patil *et al.* [15] proposed a network traffic anomaly prediction method based on principal component analysis (PCA) and a bidirectional generative adversarial network, which performs feature extraction and classification by bidirectional generative adversarial network after feature dimensionality reduction by PCA. Yu *et al.* [16] combined a convolutional neural network (CNN) and a recurrent neural network (RNN) for network traffic anomaly detection, learning data spatial features using the CNN and data temporal features using the RNN, with good detection accuracy on two datasets, namely, DARPA1998 and ISCX2012. KarunKumar Reddy [17] *et al.* investigated the advantages of DL methods over traditional ML methods for network traffic anomaly detection, and the results showed that the deep neural network structure (i.e., DL) has high accuracy. The more complex model structure of DL methods ensures their superior feature extraction capability.

Based on the above studies, traditional ML algorithms have insufficient feature learning capability, and DL methods have an oversized model that cannot be well deployed on WSNs devices. For effective traffic anomaly detection in WSNs, it is necessary to combine the shortcomings of the above studies and meet the following two requirements: first, sufficient feature learning to ensure good detection accuracy and, second, a small model size to meet the lightweight requirements of WSNs devices. Based on the above two requirements, this paper proposes a method based on PCA and a deep convolution neural network (DCNN) for DoS traffic anomaly detection in WSNs, meeting the requirements of good detection accuracy and lightweight. The contributions of this paper are as follows.

1 We propose a DL model integrating PCA and DCNN to detect the abnormal traffic of WSNs. PCA is used to reduce the dimension of the initial data, remove redundant features and extract more important features. The DCNN learns the traffic characteristics after dimensionality reduction, and classifies the normal traffic and abnormal traffic.

2 We construct a new DCNN structure to meet the requirements of lightweight and good feature extraction ability, which includes convolution layers, depthwise separable convolution layer, attention mechanism, and global average pooling (GAP). The proposed deep convolution structure has good feature learning ability and a lightweight model structure, which is more suitable for traffic anomaly detection on WSNs.

3 The proposed DCNN has a smaller model structure and higher detection rate than conventional CNN and depthwise separable convolution neural network (SNN) and performs better in terms of four classification evaluation metrics, namely, accuracy, recall, precision, and F1-score, than other mainstream network traffic anomaly detection methods, as validated on the KDDcup99, NSL-KDD, and UNSW-NB15 datasets.

This paper is organized as follows. Chapter 2 provides an introduction to the research related to network traffic anomaly detection and lightweight network. Chapter 3 introduces the three methods, namely PCA, DCNN, and attention mechanism; describes the dataset; and finally, derives the overall structure of the model. Chapter 4 conducts classification experiments on each dataset, evaluates the classification results, and finally provides a comparative assessment of the lightness of the models. Chapter 5 presents a summary, followed by an outlook on future work. Table 1 shows the abbreviations used in this paper.

TABLE 1. Summary of abbreviations.

Abbreviations	Description
WSNs	Wireless Sensor Networks
DoS	Denial of Service
ROC	Receiver Operating Characteristic
DDoS	Distributed Denial of Service
ML	Machine Learning
DL	Deep Learning
PCA	Principal Component Analysis
DCNN	Deep Convolution Neural Network
SNN	Depthwise Separable Convolutional Neural Network
CVAE	Conditional Variational Autoencoder
RF	Random Forest
SVM	Support Vector Machine
CNN	Convolution Neural Network
RNN	Recurrent Neural Network
GAP	Global Average Pooling
KNN	K-Nearest Neighbor
DT	Decision Tree
BPNN	Back Propagation Neural Networks
SDN	Software-Defined Networks
LSTM	Long Short-Term Memory
LR	Logistic Regression
Bi-LSTM	Bidirectional Long Short-Term Memory
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
AUC	Area Under ROC Curve

II. RELATED WORK

Network attacks have continued to occur in recent years, and the detection of network abnormal traffic has always been studied. At this stage, the main research direction in recent years has been focused on ML and DL.

ML-based network traffic anomaly detection has been evolving, and several studies on DoS detection using a single ML algorithm have been conducted. Adel, *et al.* [18] studied the use of SVM to detect DoS attacks. Alharbi *et al.* [19] used K-nearest neighbor (KNN) for DoS detection. Wazirali *et al.* [20] conducted an experimental comparison of various ML techniques and evaluated each method. They all achieved good results. However, as network attacks have become more diverse, a single ML technique for DoS detection has become insufficient, and other techniques such as feature selection and feature learning are now combined with ML techniques for DoS detection. Ahmad *et al.* [21] used a decision tree (DT) to detect DoS attacks; they combined the DT with a feature selection technique before classification to ensure good detection accuracy. Kiran Varma *et al.* [22] used a whale optimization algorithm to reduce feature dimensionality after using a joint RF algorithm for classification, outperforming a single detection algorithm in terms of detection accuracy. Mihoub *et al.* [23] used the concept of ’ Look-Back ’ and RF to detect DoS attacks and used the attack list detected before to strengthen the feature learning ability of the model. Finally, compared with a variety of ML and DL methods, it achieved better results. ML methods are also combined with unsupervised learning methods in network abnormal traffic detection. Drăgoi *et al.* [24] used unsupervised learning methods to analyze the distribution transfer of network abnormal traffic data and used a variety of machine learning methods to test the model. The final results show that the appropriate solution to the problem of data distribution transfer can improve the performance of the model.

Although ML algorithms achieve good network traffic anomaly detection results, the rapid change of network data and the large data volume have caused more attention to be given to DL techniques. Yue *et al.* [25] combined backpropagation neural networks (BPNN) in software-defined networks (SDN) for DoS detection, achieving a detection rate. Shi and Shen [26] proposed an unsupervised network anomaly traffic detection method based on an artificial immune network (UADAIN), which has a better detection effect than unsupervised methods such as K-means clustering. With the rapid advancement of CNN and RNN in recent years, they have been used in network traffic anomaly detection. Wu *et al.* [27] used a binary recurrent convolution method based on a coherent detection method for low-rate DoS attacks, achieving high detection performance. SaiSindhuTheja and Shyam [28] used the opposing crow search algorithm (OCSA) for feature selection and an RNN for classification. This combination of feature selection and DL classification methods achieved good results. Polat *et al.* [29] combined long short-term memory (LSTM) and gated recurrent units for DDoS feature extraction and classification. This combination of multiple DL algorithms has also been continuously investigated in recent years. The method of combining multiple algorithms based on deep learning models has also been studied. Kopp [30] Using a convGRU-based autoencoder method for unsupervised learning of network abnormal traffic, a trained autoencoder can effectively detect abnormal traffic. Duan *et al.* [31] used wavelet transform and residual learning to construct residual features in network anomaly traffic detection, used a multi-layer autoencoder to calculate the error vector, and finally learned the error through the residual network to give abnormal traffic classification results.

With the continuous development of DL-based network traffic anomaly detection, its drawbacks of large model size and many parameters have gradually emerged. Al-Turaiki and Altwaijry [32] designed a network intrusion detection system that combines a two-step data preprocessing method with a DL model. The two-step preprocessing method includes PCA and deep feature synthesis. By combining dimensionality reduction with feature engineering to retain more important features in the data, the classification performance of the algorithm is improved. The designed DL model is based on the CNN model, including five convolutional layers, two pooling layers, and four fully connected layers, and skip connections are added to prevent gradient disappearance. The proposed model has achieved good results in both binary classification and multi-classification, but at the same time,

TABLE 2. Comparison of some DL models in network anomaly traffic detection.

Methods	Datasets	Metrics	Year
BPNN [25]	World Cup 1998 Dataset	Precision / Recall / F1-score / Accuracy	2020
UADAIN [26]	ISCX 2012 / NSL-KDD	Accuracy / Detection rate / FAR	2022
OCSA-RNN [28]	KDD99	Precision / Recall / F-Measure / Accuracy	2021
LSTM-GRU-SVM [29]	SCADA	Accuracy / Sensitivity / Specificity / Precision / F1-score Confusion matrix	2022
ConvGRU- AE [30]	SWaT Dataset	Precision / Recall / FP / F1-score	2022
MSRC [31]	KDD99 / NSL-KDD UNSW-NB15 / CICIDS2018	Precision / Recall / FP / F1-score / ROC	2022
CNN-DFS [32]	NSL-KDD / UNSW-NB15	Precision / Accuracy / DR / F-measure	2021
B-Stacking [33]	CICIDS2017 / NSL-KDD	Precision / Recall / F1-score / Accuracy / ROC Confusion matrix	2022
CNNs [34]	ISCX2012 / CICIDS2017 CICIDS2018	Precision / Recall / F1-score / Accuracy / FP	2020
CNN-UAE-VAE [36]	SWaT / BATADAL / WADI	Precision / Recall / F1-score	2021
LNN [37]	UNSW-NB15 / Bot-IoT	Precision / Recall / F1-score / Accuracy / Confusion matrix	2021
DCNN	KDD99 / NSL-KDD UNSW-NB15	Precision / Recall / F1-score / Accuracy / ROC Confusion matrix	2022

its multi-layer traditional convolution structure and multiple fully connected layers also cause the model to have more parameters and cannot be well applied to some abnormal traffic detection devices with small memory. Although the computing power of today's computers is being strengthened to support the requirements of excessive computation of deep networks, lightweight models are still needed in some specific environments, such as mobile devices and wireless sensors. Some progress has been made in lightweight traffic anomaly detection, and reducing the dimensions of input data can effectively reduce the model size and increase the computational rate. For instance, Roy *et al.* [33] reduced the training time by removing multicollinearity, sampling, and dimensionality reduction of input data to make the model more lightweight. For lightweight models, not only should the input data be processed, but also the model should be simplified. DoriguzziCorin *et al.* [34] and McCullough *et al.* [35] detected DDoS attacks by building more simplified CNN models, and the results show that the simplified models have fewer parameters and shorter training time than ordinary CNN models. Thus, dimensionality reduction of input data combined with model simplification has become a general lightweight network approach. Kravchik *et al.* [36] constructed a lightweight network attack detection model using a one-dimensional (1D) CNN and an autoencoder and applied PCA to the input data for dimensionality reduction. Zhao *et al.* [37] simplified a con-based CNN model by extending and compressing its structure, improving the model's feature extraction capability using a residual inverse structure and channel shuffling operation, and performed dimensionality reduction on the initial data by PCA, resulting in high accuracy with a smaller model structure.

Because DL methods can fully extract and classify the data [38], it has great potential in DoS detection. In this paper, combined with previous research, a network anomaly traffic detection model based on PCA combined with DCNN is proposed. Based on meeting the high detection rate of DL model, the model is lightweight. Table 2 shows the comparison between the network anomaly traffic detection method of the DL model introduced above and the method proposed in this paper.

III. METHODOLOGY

A. PCA

PCA is a feature extraction method that reduces the dimensionality of data by extracting features from the original data while retaining variance information in the original data. The core idea is to achieve dimensionality reduction by calculating the correlation between data points and removing data points with high correlation. By applying dimensionality reduction, several data attributes can be converted into a few data attributes without losing substantial important information [39], and the influence of redundant information can be reduced in subsequent feature learning. Furthermore, the reduction of the number of input features reduces the number of model parameters, improving computational efficiency. The main steps of PCA are as follows.

1 Standardization of data: The standardization process normalizes the size of all data points to the same range to avoid some oversized data points causing large errors. The standardization formula is as follows.

$$X_{new} = \frac{X_i - \mu}{\sigma} \tag{1}$$

2 Calculation of covariance matrix: The correlation between data points is obtained by calculating the covariance matrix of the data points. The covariance matrix is as follows.

$$\mathrm{Cov} = \begin{bmatrix} \mathrm{Cov}_{11} & \mathrm{Cov}_{12} & \cdots & \mathrm{Cov}_{1M} \\ \mathrm{Cov}_{21} & \mathrm{Cov}_{22} & \cdots & \mathrm{Cov}_{2M} \\ \vdots & \vdots & \cdots & \vdots \\ \mathrm{Cov}_{M1} & \mathrm{Cov}_{M2} & \cdots & \mathrm{Cov}_{MM} \end{bmatrix} \tag{2}$$

3 Calculate the eigenvalues of the covariance matrix and the corresponding eigenvectors: this method is used to determine the principal components in the data.

4 Ranking selection of principal components: Feature vectors are arranged in rows from top to bottom according to the magnitude of feature values to form a vector matrix, and the order from top to bottom indicates the decreasing importance.

5 Dimension of data after dimensionality reduction: When the data dimension is to be reduced to n dimensions, i.e., the first n rows of the directional volume matrix form a new matrix.

B. DCNN

CNN [40] is widely used as a powerful feature extraction tool for target recognition, text classification, and anomaly detection. The traditional CNN mainly consists of three layers: convolution, pooling, and fully connected layers [41]; the SNN [42] is obtained by improving the convolution layer based on the traditional CNN.

The convolution layer is the most important in the CNN structure, and its function is to perform feature extraction on the target to obtain more advanced features by convolving local regions. Depthwise separable convolution consists of depthwise convolution and pointwise convolution, which compute channel and spatial features separately and then add them together, reducing the number of parameters and computational effort. The formula is as follows. Figure 1 shows the comparison between traditional convolution and depthwise separable convolution. The upper half of the figure is the traditional convolution, and the bottom half is the depthwise separable convolution. The traditional convolution is calculated using $C_1 \times n \times n \times C_2$ parameters, whereas the depthwise separable convolution is calculated separately using depthwise convolution and pointwise convolution, which only requires $C_1 \times n \times n + C_1 \times 1 \times 1 \times C_2$ parameters, significantly reducing the number of parameters.

$$\mathrm{Depthwise} - \mathrm{Conv}(W, y)_{(i,j)} = \sum\nolimits_{k,l}^{K,L} W_{(k,l)} \cdot y_{(i+k,j+l)} \tag{3}$$

$$\mathrm{Pointwise} - \mathrm{Conv}(W, y)_{(i,j)} = \sum\nolimits_{m}^{M} W_{(m)} \cdot y_{(i,j,m)} \tag{4}$$

The pooling layer follows the convolutional layer, and its main role is to compress the feature map generated by the convolutional layer and extract the main features of each local region. Pooling is divided into two methods: maximum pooling, which outputs the maximum value of each region, and average pooling, which calculates the average value of the region. The pooling method can reduce the number of features and the complexity of the network structure. However, pooling will reduce the number of original features, and there will be less repetition among the network traffic features. Some important features may be lost through pooling, affecting the accuracy of the overall network. Therefore, in this paper, we replace the pooling layer with an attention mechanism to extract the main features, thereby avoiding the loss of relevant features.

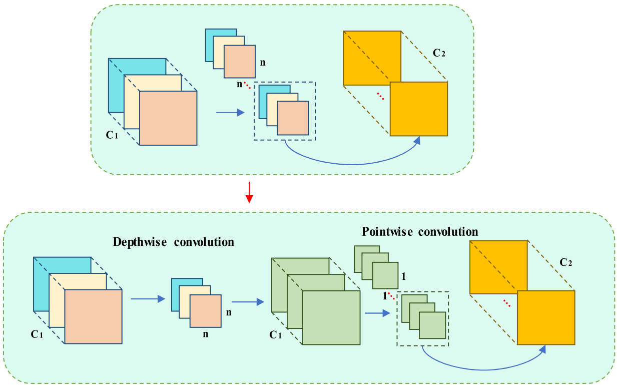


FIGURE 1. Traditional convolutional structure and depthwise separable convolutional structure.

The fully connected layer acts on the pooling layer to achieve the weighted classification of features by mapping the previously extracted features to the sample space. However, the fully connected layer has many parameters, which significantly increases the complexity of model construction. After the last convolution, the shape of the feature map input to the fully connected layer is $W \times H \times C$. When the fully connected layer has N neurons, the fully connected layer will contain $W \times H \times C \times N$ parameters, and this huge number of parameters increases the size of the model and affects computational efficiency.

GAP is a good alternative to fully connected layers [43], and Figure 2 illustrates the process of classifying fully connected layers with GAP. GAP obtains M feature maps of shape $W \times H$ after the last convolution, where M represents the number of categories. The final classification result is obtained by averaging each feature map and converting it into a 1D vector. GAP has no parameter settings, which avoids the risk of overfitting while reducing the redundant parameters of the fully connected layer. The number of parameters is reduced by replacing the fully connected layer with GAP to improve computational efficiency.

C. ATTENTION MECHANISM

The development of the attention mechanism [44] in ML was inspired by the human visual mechanism, and when humans observe a group of things, they always focus on the important parts, and this method of observation improves the efficiency of observation and helps capture the important information accurately. The attention mechanism is

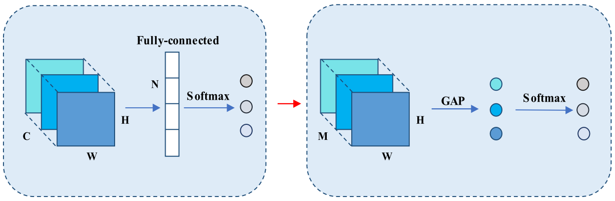


FIGURE 2. Fully connected layer structure and GAP structure.

used in practice to assign different weights to different features according to their importance among features. Greater weights are assigned to features with high importance so that the important information can be better captured during feature learning.

The attention mechanism embodies different functions depending on the position it is located. By replacing the pooling layer with the attention mechanism in the DCNN, feature loss caused by pooling downsampling is avoided; the main features are extracted by finding the more important features and assigning greater weights to each feature map after convolution by performing feature learning separately. The inclusion of the attention mechanism improves the DCNN's feature learning capability, which proved effective in subsequent experiments. The output formula is as follows.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

D. DATASET DESCRIPTION

1) DATASET INTRODUCTION

Three datasets, namely, KDDcup99 [45], NSL-KDD [46], and UNSW-NB15 [47], were used for the experiments. The KDDcup99 dataset is a classic network intrusion detection dataset containing several DoS attack data points, which is very suitable for DoS attack detection experiments. The UNSW-NB15 dataset was created by the Australian National Security Center in 2015, which has more types of attacks on the current network and better reflects the actual situation of the current network. As shown in Table 3, there are 41 feature types in the KDDcup99 and NSL-KDD datasets and 47 feature types in the UNSW-NB15 dataset; however, some of them are redundant, too repetitive with the others, or have insignificant effective features for the labels, so dimensionality reduction is needed to reduce the number of features and reduce redundancy. The three datasets, namely, KDDcup99, NSL-KDD, and UNSW-NB15, are divided into two parts, i.e., training and testing datasets, where the KDDcup99 and NSL-KDD datasets contain four attack types, and the UNSW-NB15 dataset contains nine attack types. In this paper, only their DoS attack types are extracted and studied, and Table 4 shows the number of normal and DoS attacks in their training and testing datasets.

2) DATASET PREPROCESSING

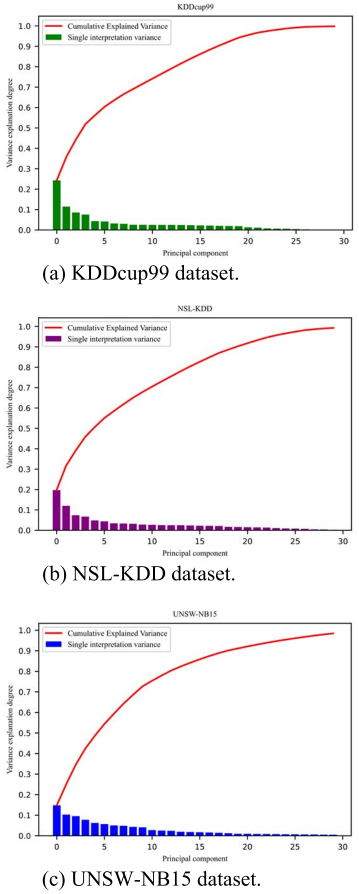
Firstly, we divide the training dataset, in which 90 % is the training set, and 10 % is the validation set. Since the model only allows numerical data input, data features are numerically encoded to convert nonnumerical features into numerical features. After numerical encoding, the normalization operation is performed in preparation for the subsequent PCA dimensionality reduction. PCA is an unsupervised dimensionality reduction method. Before dimensionality reduction, the feature items of the training dataset and testing

TABLE 3. Feature list of KDDcup99, NSL-KDD, and UNSW-NB15 dataset.

KDDcup99 and NSL-KDD		UNSW-NB15	
duration	serror_rate	srcip	trans_depth
protocol_type	srv_error_rate	sport	res_bdy_len
service	rerror_rate	dstip	sjit
flag	srv_rerror_rate	dsport	djit
src_bytes	same_srv_rate	proto	stime
dst_bytes	diff_srv_rate	state	ltime
land	srv_diff_host_rate	dur	sintpkt
wrong_fragment	dst_host_count	sbytes	dintpkt
urgent	dst_host_srv_count	dbytes	tcprtt
hot	dst_host_same_srv_rate	sttl	synack
num_failed_logins	dst_host_diff_srv_rate	dttl	ackdat
logged_in	dst_host_same_src_port_rate	sloss	is_sm_ips_ports
num_compromised	dst_host_srv_diff_host_rate	dloss	ct_state_ttl
root_shell	dst_host_serror_rate	service	ct_flw_http_mthd
su_attempted	dst_host_srv_serror_rate	sload	is_ftp_login
num_root	dst_host_rerror_rate	dload	ct_ftp_cmd
num_file_creations	dst_host_srv_rerror_rate	spkts	ct_srv_src
num_shells		dpkts	ct_srv_dst
num_access_files		swin	ct_dst_ltm
num_outbound_cmds		dwin	ct_src_ ltm
is_host_login		stcpb	ct_src_dport_ltm
is_guest_login		dtcpb	ct_dst_sport_ltm
count		smeansz	ct_dst_src_ltm
srv_count		dmeansz	

TABLE 4. KDDcup99, NSL-KDD, and UNSW-NB15 dataset categories.

Dataset	Category	Type of attack	
		Training dataset	Testing dataset
KDDcup99	Normal	97277	60592
	DoS	391458	229853
NSL-KDD	Normal	67342	9711
	DoS	45927	7635
UNSW-NB15	Normal	55999	36999
	DoS	12264	4089



(a) KDDcup99 dataset.

(b) NSL-KDD dataset.

(c) UNSW-NB15 dataset.

FIGURE 3. Dimensionality reduction effect.

dataset need to be merged to maintain the data distribution of training set, validation set, and testing set is the same after dimensionality reduction. In the dimensionality reduction stage, we reduce the features of KDDcup99, NSL-KDD, and UNSW-NB15 datasets to 30 dimensions, and the effect of dimensionality reduction is depicted in Figure 3. The histogram represents the degree of variance explained by each principal component, and the curve represents the degree of cumulative interpretation. From the figure, 30 principal components were extracted by PCA, and their cumulative principal component variance explained approximately 100% of all features, which contains a large amount of original feature information, indicating that the dimensionality reduction was successful. After dimensionality reduction, numerical normalization is also required to enable good model training, speed up convergence, and prevent gradient explosion. The normalization is given by the following equation.

$$X_{\text{norm}} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{6}$$

E. MODEL STRUCTURE

Figure 4 shows the deep convolution structure constructed in this paper, which consists of convolution layers, depthwise separable convolution layers, attention mechanism layer, and GAP layer. Firstly, we combine the traditional convolution layer with the depthwise separable convolution. In the initial layers, the traditional convolution is used to strengthen the feature learning ability, and in the subsequent layers, the depthwise separable convolution can greatly reduce the model parameters. Secondly, we use the attention mechanism to replace the pooling layer after each convolution. The multiple pooling during deep convolution operations leads to the loss of important features in the network traffic information. By replacing the pooling layer with an attention mechanism, the main features are extracted and the loss of important features is avoided. Moreover, the attention mechanism only adds a small amount of parameters, which does not affect the lightweight design of the model. Finally, GAP is used to replace the fully connected layer. The fully connected layer has complex redundant parameters, which greatly increases the complexity of the model. The replacement of GAP makes the model lighter.

Figure 5 shows the overall structure of the proposed model. The model is divided into two parts: data preprocessing and feature extraction and classification. The first part is data preprocessing. The training data and testing data should be processed through four steps: numericalization, standardization, PCA dimensionality reduction, and normalization. The preprocessing operations in each step are prepared for the next step. The second part is the feature extraction and classification of data, which is an end-to-end model. Firstly, the proposed DCNN is used to extract the feature of the preprocessed data, and then the softmax function is used to classify and evaluate the classification results of the testing set. The operation process of the proposed model is illustrated by Algorithm 1.

IV. EXPERIMENT AND RESULTS

A. IMPLEMENTATION

In this paper, simulation experiments are conducted with three datasets, namely KDDcup99, NSL-KDD, and UNSW-NB15. The experimental environment is set up on a personal host, and Table 5 presents the overall configuration.

We selected four classification evaluation metrics: accuracy, recall, precision, and F1-score. The four classification metrics can evaluate the model's performance in many aspects to avoid errors caused by a single evaluation index. The formulas of the evaluation metrics are as follows, which mainly consist of four indicators: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP is a positive case for both model prediction and the target sample's true category; TN is a negative case for both model prediction and the target sample's true category; FP is a positive case for model prediction but a negative case for the target sample's true category; FN is a negative case for model prediction but a positive case for the target sample's true category. From the formula, accuracy is the proportion of correct predictions among all predictions, which is typically

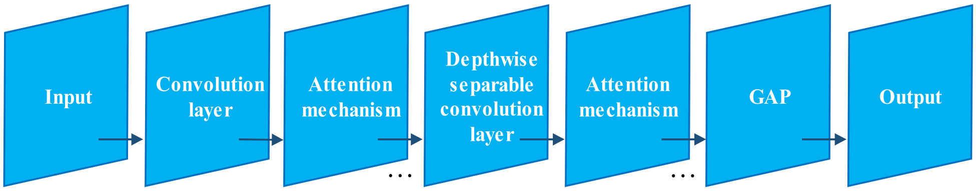


FIGURE 4. The Proposed DCNN model.

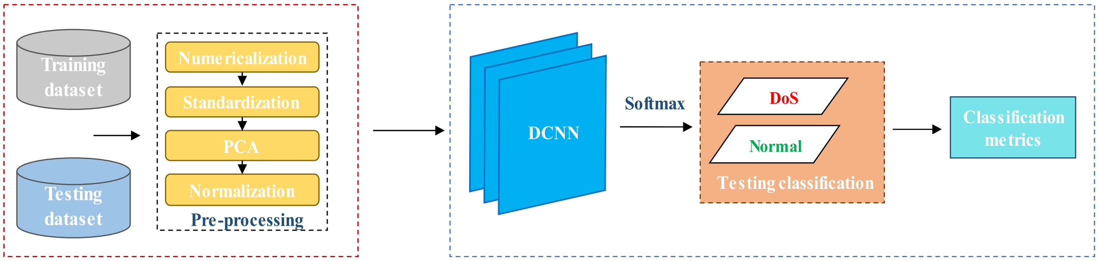


FIGURE 5. The framework of the proposed model.

```
Algorithm 1 The Computational Flow of the Proposed Model
    Input: Training dataset, Testing dataset
    Output: Classification metrics
Extract Features (x) and Labels (y) from Training
  dataset and Testing dataset
while data pre − processing do
    Perform numerical encoding;
    Standardization of features;
    Feature reduction with PCA;
    Normalized features;
for i in {1, 2, ..., n} do
    Load DCNN model
    Input Training dataset to DCNN model
    Calculate training loss
    Backpropagation update weight
    Save the updated model
    Repeat until the cycle is complete
Testing model with Testing dataset
Calculate Classification metrics
end
```

TABLE 5. Experimental operating environment.

Project	Environmental Parameters
CPU	Intel core i5 10600KF
GPU	NVIDIA RTX2060
Python version	3.6
TensorFlow version	1.14
Keras version	2.2.5

used as an indicator to evaluate the overall classification performance of a model; recall is the proportion of positive predictions among the true categories of samples, which is more important than accuracy in abnormal traffic detection, as it reflects the recognition of abnormal traffic by a model. Precision is the percentage of correct predictions among positive predictions; F1-score is the average of recall and precision. In addition to the four evaluation metrics, we also visualize the detection effect through ROC curves and confusion matrices. The ROC curve indicates the trade-off between the TP rate and the FP rate of the estimated classes [48], and the confusion matrix is composed of TP, TN, FP, and FN. The model size and number of parameters are chosen for evaluation in terms of the lightweight property to determine how lightweight the model is.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$Recall = \frac{TP}{FN + TP} \tag{8}$$

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$F1 - score = \frac{2PR}{P + R} \tag{10}$$

Experimental simulation is divided into two parts, classification metrics analysis, and lightweight metrics analysis. In the classification comparison experiments, we used seven models, namely, logistic regression (LR), DT, KNN, LSTM, bidirectional LSTM (Bi-LSTM), CNN, and SNN, for comparison with the DCNN model. It includes the classical ML method and the current mainstream DL method to verify the effectiveness of the model. In the lightweight evaluation of the model, we compare the proposed model with the original CNN and SNN, and Table 6 shows their model structures. All

TABLE 6. The respective model parameters of CNN, SNN, and DCNN.

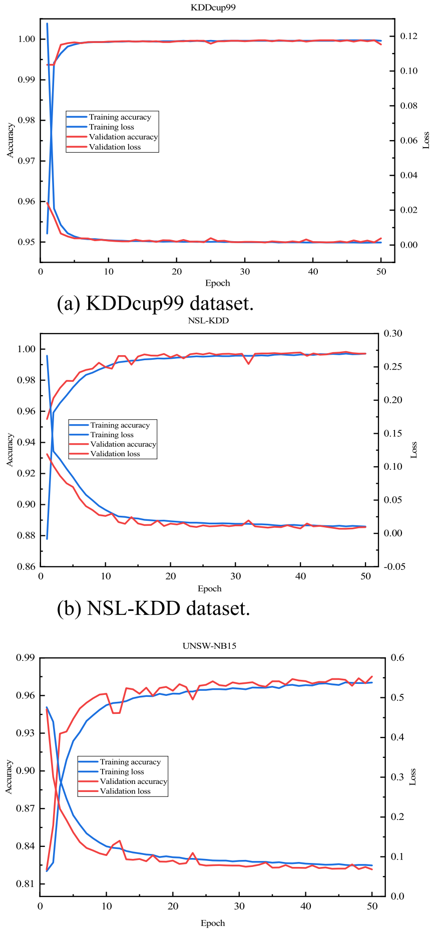
CNN	SNN	**DCNN**
Conv1D (32,3, selu)	SConv1D (32,3, selu)	Conv1D (32,3, selu)
		Attention
Conv1D (32,3, selu)	SConv1D (32,3, selu)	Conv1D (32,3, selu)
Max-pooling	Max-pooling	Attention
Conv1D (64,3, selu)	SConv1D (64,3, selu)	SConv1D (64,3, selu)
		Attention
Conv1D (64,3, selu)	SConv1D (64,3, selu)	SConv1D (64,3, selu)
		Attention
Max-pooling	Max-pooling	GAP
Fully-connected (128, selu)	Fully-connected (128, selu)	Softmax
Softmax	Softmax	

three models have a four-layer 1D convolutional structure. SELU is chosen as the activation function. SELU has the feature of self-normalization compared with ReLU, which can further prevent gradient disappearance and gradient explosion problems.

B. CLASSIFICATION METRICS ANALYSIS

Figure 6 shows the training accuracy, training loss, validation accuracy, and validation loss of the DCNN model on the three datasets, and it can be seen that with the increase of period, the accuracy and loss of the model eventually converge to the maximum and minimum values. In general, the training and validation accuracy of KDDcup99 dataset is the best, followed by NSL-KDD dataset, and the worst is UNSW-NB15 dataset. However, in terms of the detection effect of binary classification, the model can achieve satisfactory results on three datasets. This shows that the model structure design is good, and the model can effectively learn the characteristics of different datasets through training.

Figure 7 shows the ROC curve of different models on three datasets. The ROC curve is composed of FP rate and TP rate. When the Area Under ROC Curve (AUC) is larger, it represents the better performance of the model. It can be seen that on the KDDcup99 dataset, each model has a good performance, and the best performance is CNN, followed by DCNN. On NSL-KDD and UNSW-NB15 datasets, DCNN performed the best, with AUCs of 0.945 and 0.993, respectively. On the whole, the performance of the DL model on the three data sets is better than that of the ML model, which is due to the more stable feature learning ability of the DL model. The ML model is more superficial in feature learning, and it is difficult to achieve good results in the face of complex data distribution. At the same time, it can be seen that the AUC of the NSL-KDD dataset is much lower than that of the KDDcup99 dataset and the UNSW-NB15 dataset, which is related to the distribution of the NSL-KDD dataset. Its test set contains more feature distributions that the training set does not have, which is more to test the generalization performance of the model.



(a) KDDcup99 dataset.

(b) NSL-KDD dataset.

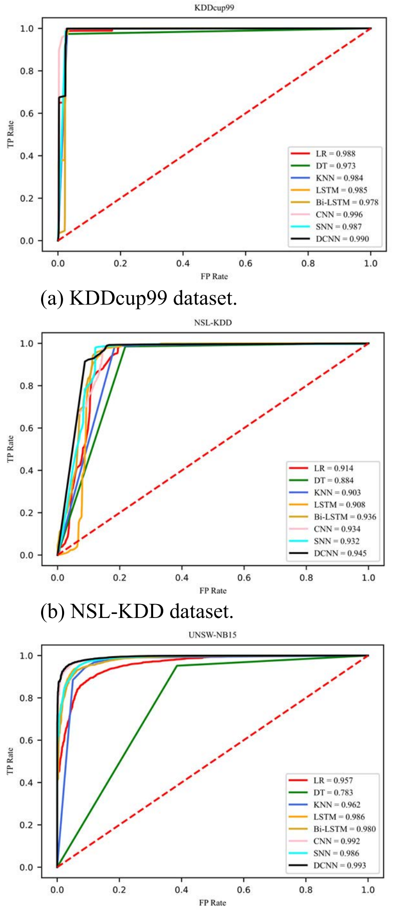
(c) UNSW-NB15 dataset.

FIGURE 6. Accuracy and loss of training set and validation set.

TABLE 7. Different methods on the KDDcup99 dataset.

Method	Accuracy	Recall	Precision	F1-score
LR	97.02%	97.02%	97.29%	97.15%
DT	97.22%	97.22%	97.37%	97.29%
KNN	97.54%	97.54%	97.78%	97.66%
LSTM	97.32%	97.32%	97.62%	97.47%
Bi-LSTM	97.38%	97.38%	97.66%	97.52%
CNN	97.52%	97.52%	97.79%	97.65%
SNN	97.42%	97.42%	97.70%	97.56%
DCNN	**97.83%**	**97.83%**	**98.02%**	**97.92%**

Tables 7, 8, and 9 show the two classification results for the Normal category and the DoS attack category for the three datasets, namely, KDDcup99, NSL-KDD, and UNSW-NB15, with the overall evaluation metrics. From the tables, for the KDDcup99 dataset, DCNN outperforms the other methods in terms of all evaluation metrics. Nonetheless, the metrics of the other methods are all above 97%, indicating that the methods are effective in detecting the KDDcup99 dataset. For the



(a) KDDcup99 dataset.

(b) NSL-KDD dataset.

(c) UNSW-NB15 dataset.

FIGURE 7. ROC curves comparison.

TABLE 8. Different methods on the NSL-KDD dataset.

Method	Accuracy	Recall	Precision	F1-score
LR	88.99%	88.99%	90.40%	89.69%
DT	89.55%	89.55%	90.65%	90.10%
KNN	89.02%	89.02%	90.38%	89.69%
LSTM	90.04%	90.04%	91.04%	90.54%
Bi-LSTM	90.51%	90.51%	91.61%	91.06%
CNN	91.92%	91.92%	92.71%	92.31%
SNN	90.06%	90.06%	91.30%	90.68%
DCNN	**92.28%**	**92.28%**	**92.99%**	**92.63%**

NSL-KDD and UNSW-NB15 datasets, DCNN also achieves the best result. and the DL models outperform the traditional ML models, reflecting the stronger feature extraction ability of DL methods. It can also be seen that the DCNN designed in this paper has a better performance compared with CNN and SNN models, and its Accuracy improves by 0.31% and 0.41% on the KDDcup99 dataset, 0.36% and 2.22% on the NSL-KDD dataset, and 0.57% and 1.24% on the UNSW-NB15 dataset, respectively. It indicates that the model proposed in this paper has a better overall detection effect.

TABLE 9. Different methods on the UNSW-NB15 dataset.

Method	Accuracy	Recall	Precision	F1-score
LR	84.38%	84.38%	93.11%	88.53%
DT	91.83%	91.83%	92.02%	91.92%
KNN	95.87%	95.87%	96.29%	96.08%
LSTM	95.45%	95.45%	96.30%	95.87%
Bi-LSTM	93.70%	93.70%	94.17%	93.93%
CNN	96.19%	96.19%	96.87%	96.53%
SNN	95.52%	95.52%	96.32%	95.92%
DCNN	**96.76%**	**96.76%**	**97.17%**	**96.96%**

TABLE 10. Classification performance on the KDDcup99 dataset.

Method	Class	Recall	Precision	F1-score
LR	Normal	98.6%	88.4%	93.2%
	DoS	96.6%	99.6%	98.1%
DT	Normal	97.4%	90.1%	93.6%
	DoS	97.2%	99.3%	98.2%
KNN	Normal	99.7%	89.7%	94.4%
	DoS	97.0%	**99.9%**	98.4%
LSTM	Normal	**99.9%**	88.7%	94.0%
	DoS	96.6%	**99.9%**	98.3%
Bi-LSTM	Normal	99.8%	89.0%	94.1%
	DoS	96.7%	**99.9%**	98.3%
CNN	Normal	**99.9%**	89.5%	94.4%
	DoS	96.9%	**99.9%**	98.4%
SNN	Normal	**99.9%**	89.1%	94.2%
	DoS	96.8%	**99.9%**	98.3%
DCNN	Normal	99.7%	**90.8%**	**95.0%**
	DoS	**97.3%**	**99.9%**	**98.6%**

To thoroughly evaluate the proposed model's performance, we evaluated the model's classification performance for the Normal category and the DoS attack category separately in terms of the metrics, as shown in Tables 10, 11, and 12. We mainly focus on the recall of DoS categories in the model. A higher recall represents a higher correct detection rate of DoS categories. It can be seen from the table that on KDDcup99 and NSL-KDD datasets, DCNN recall is the highest, followed by CNN. On the UNSW-NB15 dataset, CNN has the highest recall, followed by DCNN. It shows that DCNN has higher detection performance for DoS attack traffic. For F1-score of normal and DoS categories, DCNN is also the best, indicating that DCNN has better feature learning ability than other models. Also overall, the DL model performs better compared to the ML model.

To more intuitively visualize how the model improvements help in the detection of DoS categories, Figures 8, 9, and 10 show the classification confusion matrices of CNN, SNN, and DCNN on the three datasets, namely, KDDcup99, NSL-KDD, and UNSW-NB15. From the confusion matrices, it can be seen that compared with CNN and SNN, the number of correct DoS detections of DCNN on KDDcup99 dataset increased by 1029 and 1342, respectively, and the number of correct DoS detections on NSL-KDD dataset increased by 61 and 393, respectively. For the UNSW-NB15 dataset, the

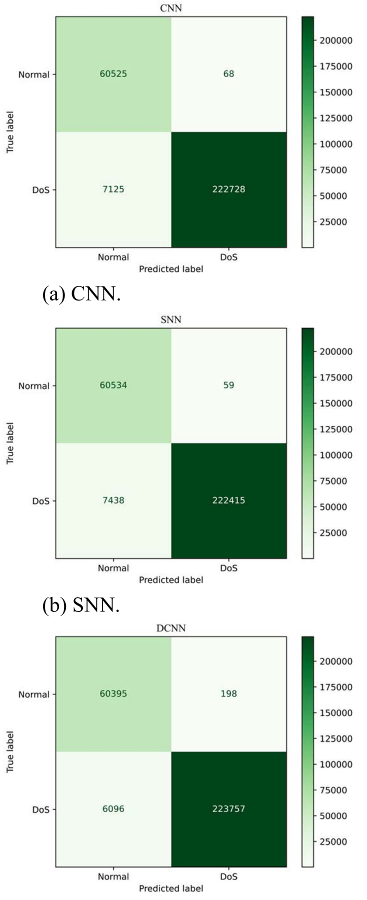
TABLE 11. Classification performance on the NSL-KDD dataset.

Method	Class	Recall	Precision	F1-score
LR	Normal	99.0%	84.1%	91.0%
	DoS	76.3%	98.4%	85.9%
DT	Normal	98.5%	85.2%	91.3%
	DoS	78.2%	97.6%	86.8%
KNN	Normal	98.9%	84.2%	91.0%
	DoS	76.5%	98.2%	86.0%
LSTM	Normal	98.6%	85.8%	91.7%
	DoS	79.2%	97.7%	87.5%
Bi-LSTM	Normal	99.2%	86.0%	92.1%
	DoS	79.4%	98.8%	88.0%
CNN	Normal	99.2%	87.9%	93.2%
	DoS	82.6%	98.8%	90.0%
SNN	Normal	**99.3%**	85.3%	91.8%
	DoS	78.3%	**98.9%**	87.4%
DCNN	Normal	99.2%	**88.4%**	**93.5%**
	DoS	**83.4%**	98.8%	**90.5%**

TABLE 12. Classification performance on the UNSW-NB15 dataset.

Method	Class	Recall	Precision	F1-score
LR	Normal	83.4%	99.2%	90.6%
	DoS	93.6%	38.3%	54.4%
DT	Normal	95.2%	95.7%	95.5%
	DoS	61.5%	58.5%	60.0%
KNN	Normal	96.7%	98.7%	97.7%
	DoS	88.2%	74.8%	81.0%
LSTM	Normal	95.8%	99.1%	97.4%
	DoS	92.2%	70.9%	80.1%
Bi-LSTM	Normal	95.7%	97.3%	96.5%
	DoS	75.9%	65.9%	70.6%
CNN	Normal	96.4%	**99.4%**	97.9%
	DoS	**94.5%**	74.2%	83.1%
SNN	Normal	95.9%	99.1%	97.5%
	DoS	92.0%	71.3%	80.3%
DCNN	Normal	**97.1%**	99.3%	**98.2%**
	DoS	93.5%	**78.2%**	**85.2%**

number of correct detections of DoS is reduced by 38 and increased by 62 for DCNN compared to CNN and SNN, respectively. Because SNN adopts the structure of separable convolution, its feature extraction ability is not as good as CNN composed of traditional convolution structure. It can also be seen from the confusion matrix that the detection rate of SNN for DoS is lower than CNN on all three data sets. DCNN adopts the combination of traditional convolution and separable convolution, and replaces the maximum pooling layer with attention mechanism. Its feature extraction ability can reach the same level as CNN, so the detection rate of DoS on three data sets is similar to CNN.



(a) CNN.

(b) SNN.

(c) DCNN.

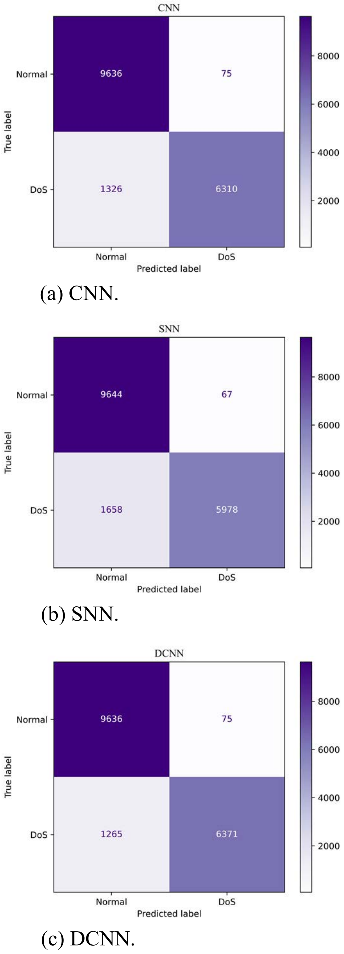
FIGURE 8. Confusion matrices on the KDDcup99 dataset.

C. LIGHTWEIGHT METRIC ANALYSIS

In this section, we compare the degree of lightness of DCNN with CNN and SNN. In the lightweight comparison, the model size and the number of parameters were mainly chosen to measure whether the model has an appropriate size for deployment on WSNs devices, and Table 13 shows the comparison results. For the three datasets, DCNN reduces the size of CNN and SNN models by 87.46% and 84.77%, respectively, and the number of parameters is reduced by 69496 and 55483 compared with CNN and SNN models. The reduction of model parameters is mainly due to the use of separable convolution and GAP.DCNN introduces separable convolution and GAP into the traditional convolution structure to reduce the overall calculation of the model and has a smaller model size. Experiments demonstrate that DCNN is more suitable for WSNs devices with a small memory footprint.

D. DISCUSSION AND ANALYSIS

In this chapter, the DCNN model is used to detect and analyze DoS traffic in KDDcup99, NSL-KDD, and UNSW-NB15 datasets. Compared with other models, DCNN has better experimental results, which proves the superiority of DCNN

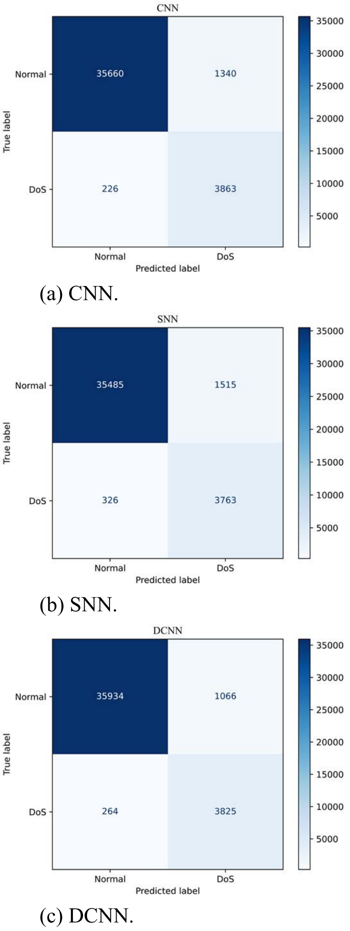


(a) CNN.

(b) SNN.

(c) DCNN.

FIGURE 9. Confusion matrices on the NSL-KDD dataset.



(a) CNN.

(b) SNN.

(c) DCNN.

FIGURE 10. Confusion matrices on the UNSW-NB15 dataset.

TABLE 13. Compare model size and number of parameters.

Method	Dataset	Model size	Parameters
CNN	KDDcup99 NSL-KDD UNSW-NB15	311KB	79522
SNN	KDDcup99 NSL-KDD UNSW-NB15	256KB	65509
DCNN	KDDcup99 NSL-KDD UNSW-NB15	**39KB**	**10026**

model in detecting abnormal traffic on WSNs devices. The following information is also available.

(1) The NSL-KDD dataset has a worse detection effect than the KDDcup99 and UNSW-NB15 datasets. This is because the test set has more unknown data features than the training set. This feature makes it possible to better test the generalization of the model.

(2) Although separable convolution can reduce model parameters, its feature extraction ability is also worse than traditional convolution. The combination of traditional convolution and separable convolution has better feature extraction ability, and using attention mechanism to replace the maximum pooling layer can also make up for the lack of feature extraction ability of separable convolution.

(3) Separable convolution and GAP have fewer parameters than traditional convolution and fully connected layers, which benefits from their less computation and simplifies the model operation process.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed an effective traffic anomaly detection method given that WSNs are vulnerable to attacks and their devices have limited storage space. First, PCA was used for data dimensionality reduction to eliminate redundancy and reduce the number of parameters for subsequent model training. A new DCNN structure is constructed to meet the requirements of lightweight and high detection ability of the model. On the construction of DCNN, the traditional convolution layer and depthwise separable convolution layer are combined, and the pooling layer was replaced by attention mechanism for feature extraction while avoiding the feature loss caused by pooling. Finally, the GAP is used to replace the fully connected layer to reduce the number of parameters of the model. The model was validated on three datasets, namely, KDDcup99, NSL-KDD, and UNSW-NB15. The results showed that the proposed model achieved better ROC curves performance, and has a higher correct detection number for DoS attacks than other ML and DL models. It also has the same feature extraction capability as CNN while reducing the model size compared to CNN and SNN. We provide a feasible scheme for traffic anomaly detection in WSNs. In future work, we aim to make the model more lightweight and to investigate more novel network attacks.

REFERENCES

[1] P. K. Chithaluru, M. S. Khan, M. Kumar, and T. Stephan, ''ETH-LEACH: An energy enhanced threshold routing protocol for WSNs,'' *Int. J. Commun. Syst.*, vol. 34, no. 12, 2021, Art. no. e4881.

[2] R. Zhao, J. Yin, Z. Xue, G. Gui, B. Adebisi, T. Ohtsuki, H. Gacanin, and H. Sari, ''An efficient intrusion detection method based on dynamic autoencoder,'' *IEEE Wireless Commun. Lett.*, vol. 10, no. 8, pp. 1707–1711, Aug. 2021.

[3] M. S. Abdalzaher and O. Muta, ''A game-theoretic approach for enhancing security and data trustworthiness in IoT applications,'' *IEEE Internet Things J.*, vol. 7, no. 11, pp. 11250–11261, Nov. 2020.

[4] P. Chithaluru, F. Al-Turjman, and T. Stephan, ''Energy-efficient blockchain implementation for cognitive wireless communication networks (CWCNs),'' *Energy Rep.*, vol. 7, pp. 8277–8286, Nov. 2021.

[5] M. Huang, K. Ding, and S. Dey, ''Learning-based DoS attack power allocation in multiprocess systems,'' *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 15, 2022, doi: 10.1109/TNNLS.2022.3148924.

[6] M. Barati, A. Abdullah, N. I. Udzir, R. Mahmod, and N. Mustapha, ''Distributed denial of service detection using hybrid machine learning technique,'' in *Proc. Int. Symp. Biometrics Secur. Technol. (ISBAST)*, Aug. 2014, pp. 268–273.

[7] P. A. R. Kumar and S. Selvakumar, ''Detection of distributed denial of service attacks using an ensemble of adaptive and hybrid neuro-fuzzy systems,'' *Comput. Commun.*, vol. 36, no. 3, pp. 303–319, Feb. 2013.

[8] Radware. (2018). *Memcached DDoS Attacks*. Accessed: Mar. 22, 2022. [Online]. Available: https://security.radware.com/ddos-threats-attacks/threat-advisories-attack-reports/memcached-under-attack/

[9] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, ''Lucid: A practical, lightweight deep learning solution for DDoS attack detection,'' *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 876–889, Jun. 2020.

[10] M. S. Abdalzaher, M. Elwekeil, T. Wang, and S. Zhang, ''A deep autoencoder trust model for mitigating jamming attack in IoT assisted by cognitive radio,'' *IEEE Syst. J.*, vol. 16, no. 3, pp. 3635–3645, Sep. 2022.

[11] M. Monshizadeh, V. Khatri, M. Gamdou, R. Kantola, and Z. Yan, ''Improving data generalization with variational autoencoders for network traffic anomaly detection,'' *IEEE Access*, vol. 9, pp. 56893–56907, 2021.

[12] Q. Ma, C. Sun, B. Cui, and X. Jin, ''A novel model for anomaly detection in network traffic based on kernel support vector machine,'' *Comput. Secur.*, vol. 104, May 2021, Art. no. 102215.

[13] S. Iranmanesh, F. S. Abkenar, A. Jamalipour, and R. Raad, ''A heuristic distributed scheme to detect falsification of mobility patterns in internet of vehicles,'' *IEEE Internet Things J.*, vol. 9, no. 1, pp. 719–727, Jan. 2022.

[14] M. S. Abdalzaher, M. S. Soliman, S. M. El-Hady, A. Benslimane, and M. Elwekeil, ''A deep learning model for earthquake parameters observation in IoT system-based earthquake early warning,'' *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8412–8424, Jun. 2021.

[15] R. Patil, R. Biradar, V. Ravi, P. Biradar, and U. Ghosh, ''Network traffic anomaly detection using PCA and BiGAN,'' *Internet Technol. Lett.*, vol. 5, no. 1, p. e235, Jan. 2022.

[16] G. Wei and Z. Wang, ''Adoption and realization of deep learning in network traffic anomaly detection device design,'' *Soft Comput.*, vol. 25, no. 2, pp. 1147–1158, Jan. 2021.

[17] D. K. Reddy, H. S. Behera, J. Nayak, P. Vijayakumar, B. Naik, and P. K. Singh, ''Deep neural network based anomaly detection in Internet of Things network traffic tracking for the applications of future smart cities,'' *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 7, p. e4121, 2021.

[18] A. Abusitta, M. Bellaiche, and M. Dagenais, ''An SVM-based framework for detecting DoS attacks in virtualized clouds under changing environment,'' *J. Cloud Comput.*, vol. 7, no. 1, pp. 1–18, Dec. 2018.

[19] Y. Alharbi, A. Alferaidi, K. Yadav, G. Dhiman, and S. Kautish, ''Denial-of-service attack detection over IPv6 network based on KNN algorithm,'' *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–6, Dec. 2021.

[20] R. Wazirali and R. Ahmad, ''Machine learning approaches to detect DoS and their effect on WSNs lifetime,'' *Comput., Mater. Continua*, vol. 70, no. 3, pp. 4922–4946, 2022.

[21] R. Ahmad, R. Wazirali, Q. Bsoul, T. Abu-Ain, and W. Abu-Ain, ''Feature-selection and mutual-clustering approaches to improve DoS detection and maintain WSNs' lifetime,'' *Sensors*, vol. 21, no. 14, p. 4821, Jul. 2021.

[22] K. V. P. Ravi, K. V. S. Raju, and R. Suresh, ''Application of whale optimization algorithm in DDOS attack detection and feature reduction,'' in *Inventive Computation and Information Technologies*, vol. 173. Singapore: Springer, 2021, pp. 93–102.

[23] A. Mihoub, O. B. Fredj, O. Cheikhrouhou, A. Derhab, and M. Krichen, ''Denial of service attack detection and mitigation for Internet of Things using looking-back-enabled machine learning techniques,'' *Comput. Electr. Eng.*, vol. 98, Mar. 2022, Art. no. 107716.

[24] M. Drăgoi, E. Burceanu, E. Haller, A. Manolache, and F. Brad, ''AnoShift: A distribution shift benchmark for unsupervised anomaly detection,'' 2022, *arXiv:2206.15476*.

[25] M. Yue, H. Wang, L. Liu, and Z. Wu, ''Detecting DoS attacks based on multi-features in SDN,'' *IEEE Access*, vol. 8, pp. 104688–104700, 2020.

[26] Y. Shi and H. Shen, ''Unsupervised anomaly detection for network traffic using artificial immune network,'' *Neural Comput. Appl.*, vol. 34, no. 15, pp. 13007–13027, Aug. 2022.

[27] Z. Wu, Y. Yin, G. Li, and M. Yue, ''Coherent detection of synchronous low-rate DoS attacks,'' *Secur. Commun. Netw.*, vol. 2021, pp. 1–14, Mar. 2021.

[28] R. SaiSindhuTheja and G. K. Shyam, ''An efficient Metaheuristic algorithm based feature selection and recurrent neural network for DoS attack detection in cloud computing environment,'' *Appl. Soft Comput.*, vol. 100, Mar. 2021, Art. no. 106997.

[29] H. Polat, M. Türkoğlu, O. Polat, and A. Şengür, ''A novel approach for accurate detection of the DDoS attacks in SDN-based SCADA systems based on deep recurrent neural networks,'' *Expert Syst. Appl.*, vol. 197, Jul. 2022, Art. no. 116748.

[30] F. Kopp, ''Representation learning for content-sensitive anomaly detection in industrial networks,'' 2022, *arXiv:2205.08953*.

[31] X. Duan, Y. Fu, and K. Wang, ''Network traffic anomaly detection method based on multi scale residual feature,'' 2022, *arXiv:2205.03907*.

[32] I. Al-Turaiki and N. Altwaijry, ''A convolutional neural network for improved anomaly-based network intrusion detection,'' *Big Data*, vol. 9, no. 3, pp. 233–252, Jun. 2021.

[33] S. Roy, J. Li, B.-J. Choi, and Y. Bai, ''A lightweight supervised intrusion detection mechanism for IoT networks,'' *Future Gener. Comput. Syst.*, vol. 127, pp. 276–285, Feb. 2022.

[34] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, and D. Siracusa, ''Lucid: A practical, lightweight deep learning solution for DDoS attack detection,'' *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 876–889, Jun. 2020.

[35] E. McCullough, R. Iqbal, and A. Katangur, ''Analysis of machine learning techniques for lightweight DDoS attack detection on IoT networks,'' in *Proc. Int. Conf. Forthcoming Netw. Sustainability IoT Era*, vol. 353, 2021, pp. 96–110.

[36] M. Kravchik and A. Shabtai, ''Efficient cyber attack detection in industrial control systems using lightweight neural networks and PCA,'' *IEEE Trans. Depend. Secure Comput.*, vol. 19, no. 4, pp. 2179–2197, Aug. 2022.

[37] R. Zhao, G. Gui, Z. Xue, J. Yin, T. Ohtsuki, B. Adebisi, and H. Gacanin, ''A novel intrusion detection method based on lightweight neural network for Internet of Things,'' *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9960–9972, Jun. 2020.

[38] M. Mittal, K. Kumar, and S. Behal, ''Deep learning approaches for detecting DDoS attacks: A systematic review,'' *Soft Comput.*, pp. 1–37, 2022.

[39] S. P. Rm, P. K. R. Maddikunta, M. Parimala, S. Koppu, T. R. Gadekallu, C. L. Chowdhary, and M. Alazab, ''An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture,'' *Comput. Commun.*, vol. 160, pp. 139–149, Jul. 2020.

[40] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, ''Gradient-based learning applied to document recognition,'' *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[41] M. S. ElSayed, N.-A. Le-Khac, M. A. Albahar, and A. Jurcut, "A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique," *J. Netw. Comput. Appl.*, vol. 191, Oct. 2021, Art. no. 103160.

[42] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.

[43] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*.

[44] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, Sep. 2021.

[45] K. Siddique, Z. Akhtar, F. A. Khan, and Y. Kim, "KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research," *Computer*, vol. 52, no. 2, pp. 41–51, Feb. 2019.

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

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

[48] M. S. Abdalzaher, S. S. R. Moustafa, M. Abd-Elnaby, and M. Elwekeil, "Comparative performance assessments of machine-learning methods for artificial seismic sources discrimination," *IEEE Access*, vol. 9, pp. 65524–65535, 2021.

CHENGPENG YAO received the Graduate degree from the School of Information Engineering, Engineering University of PAP. His research interests include network security situation awareness and deep learning.

YU YANG received the Ph.D. degree from Shenyang Agricultural University. He is currently an Associate Professor with the School of Information Engineering, Engineering University of PAP. His research interests include network security and machine learning.

KUN YIN received the Graduate degree from the School of Information Engineering, Engineering University of PAP. His research interest includes anomaly detection.

JINWEI YANG received the Graduate degree from the School of Information Engineering, Engineering University of PAP. Her research interest includes intrusion detection.