IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 23, 2019, accepted March 9, 2019, date of publication March 20, 2019, date of current version April 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905041

Network Intrusion Detection: Based on Deep Hierarchical Network and Original Flow Data

YONG ZHANG, XU CHEN, LEI JIN, XIAOJUAN WANG, AND DA GUO
School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Xu Chen (buptchenxu@gmail.com)

This work was supported by the National Natural Science Foundation of China under Grant 61601053.

ABSTRACT Network intrusion detection plays a very important role in protecting computer network security. The abnormal traffic detection and analysis by extracting the statistical features of flow is the main analysis method in the field of network intrusion detection. However, these features need to be designed and extracted manually, which often loses the original information of the flow and leads to poor detection efficiency. In this paper, we do not manually design the features of the flow but directly extract the raw data information of the flow for analysis. In addition, we first proposed a new network intrusion detection model named the deep hierarchical network, which integrates the improved LeNet-5 and LSTM neural network structures, while learning the spatial and temporal features of flow. By designing a reasonable network cascading method, we can train our proposed hierarchical network at the same time instead of training two networks separately. In this paper, we use the CICIDS2017 dataset and the CTU dataset. The number and types of flow in these two datasets are large, and the attack types are relatively new. The experimental results show that the performance of the proposed hierarchical network model is significantly better than other network intrusion detection models, which can achieve the best detection accuracy. Finally, we also present an analysis method for traffic features which has an important contribution to abnormal traffic detection and gives the actual meanings of these important features.

INDEX TERMS Network intrusion detection, deep hierarchical network, raw feature, feature importance.

I. INTRODUCTION

With the continuous expansion and rapid development of the Internet, it has brought great convenience to network users. But along with the development of the Internet, there have also been a series of attacks. A targeted attacker takes appropriate attacks against a specific network to cause the network to crash, thereby failing to provide users with safe and reliable services, resulting in huge economic losses. The task of network intrusion detection is to discover suspicious attacks [1] and take corresponding measures to protect the network from sustaining attacks and reduce economic losses. Traffic classification is an important task of network intrusion detection [2]. It requires researchers to accurately judge the collected traffic datasets and detect traffic with attack behaviors. Traffic classification mainly analyzes some key fields in the traffic packets to determine whether the network is attacked or has abnormal behaviors that violate network security. According to the classification test results of the traffic, a feedback message is sent to the network to determine whether the network needs to disconnect or give an alarm message.

In order to detect abnormal traffic efficiently, the traffic packets are usually divided into flow [3] according to the source ip, destination ip, source port, destination port, protocol, and timestamp. At present, there are mature traffic detection technologies, including port-based method, payload-based method and statistical feature-based method.

The port-based traffic detection method [4] is commonly used and effective in the early days. In the early days of the Internet, network protocols used for network traffic were relatively simple, and specific applications basically used fixed port numbers. Therefore, when an application is attacked by other applications, abnormal traffic packets can be effectively detected based on the port number. However, with the advent of dynamic port allocation technology, ports can be easily redirected. Therefore, the port-based traffic detection method cannot adequately express the traffic attributes of the network, and the traffic detection effect is often poor.

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

The payload-based traffic detection method [5], [6] uses the information of the application layer protocol to express the features of the traffic, the most representative of which is the deep packet inspection (DPI) technology [7]. Deep packet inspection technology needs to decrypt and encrypt the transmitted traffic data. By modeling and analyzing the transmitted data information, malicious traffic packets can be detected very effectively. Although the deep packet inspection technology is a widely used abnormal traffic detection technology in practical applications, with the rise of encryption protocols (such as https) and the increasing emphasis on privacy, deep packet inspection technology is no longer recommended. In addition, the use of deep packet inspection in the decryption processing of traffic is very expensive. With the rapid growth of Internet traffic, deep packet inspection technology needs to consume huge computing resources when decrypting traffic packets.

The statistical feature based traffic detection method [8] generally uses the packet arrival time, the packet size and the statistical features of the traffic packet fields (eg, average, maximum, minimum) to express the attributes of the traffic. Using these artificially designed features and machine learning algorithms to analyze and detect abnormal traffic [9] have become relatively reliable methods, but the traffic data needs to be accurately labeled when training a supervised algorithm model.

In the previous work, researchers mainly operated from the data level to improve the classification accuracy and other metrics. Whether it is traditional machine learning algorithms or various neural network algorithms in deep learning, researchers try to extract information from traffic data through complex feature engineering. Their feature engineering can extract the temporal feature and spatial feature of the flow data, but feature engineering will lose some information or change the original temporal and spatial features of the traffic packets. Yeo *et al.* [10] extracted temporal features such as fiat, biat and duration, while Yu *et al.* [11] extracted temporal features such as activation time of flow, time interval, packet arrival time and spatial features such as packet number, IP address and transmission direction. Through the traffic features they extracted, algorithms can only use the missing traffic data information to perform classification, so the classification accuracy and other metrics have reached the bottleneck and can hardly continue to improve.

This paper uses the deep learning method in the field of machine learning to classify flow. The neural network model in deep learning can automatically extract features from the input data for training. It has good self-adaptation, self-organization and promotion ability to make the system have higher detection efficiency. The proposed method only uses the original information of traffic data as the features of flow, and uses the hierarchical network structure to automatically learn the spatial and temporal features of flow without complex feature engineering. By analyzing the experimental results, we find that the spatial and temporal features extracted by the separate CNN and LSTM models have similar shortcomings compared with the feature engineering. The data does contain rich features with classification recognition capabilities, but since the separate CNN and LSTM only utilize the spatial feature or temporal feature of flow respectively, this is equivalent to discarding some information. So if we want to further improve the classification accuracy and other metrics, we need to extract the spatial and temporal features of the flow simultaneously using a hierarchical network. Code has been released at *https://github.com/chenxu93/abnormal-traffic*. The main contributions of this paper are as follows:

(1) We propose a new method for extracting flow features, which preserves all the information of the flow as much as possible. The flow features we extracted do not require any prior knowledge, so we don't need to manually extract the flow features with specific meanings.

(2) For the first time, we propose a new deep hierarchical network model structure to learn their temporal features and spatial features simultaneously from the original flow data. Our model achieves the best performance on all metrics.

(3) We propose a method to analyze the flow features, which can find the features that contribute significantly to abnormal flow detection and we give the true meanings of these important features.

The structure of this paper is as follows. Section II describes some of the related work of network intrusion detection. Section III details the abnormal flow classification detection model we used in this paper. In section IV, we describe the two datasets used in this paper and show the experimental results we performed on the two datasets. In section V, we analyze the flow features that have important contributions to abnormal flow detection. Finally, Section VI gives a conclusion of this article.

II. RELATED WORKS

The concept of intrusion detection technology was first proposed by Anderson [12] in 1980, with the goal of identifying anomalous behaviors in the network. Reduce the losses of the network by taking appropriate measures against abnormal behaviors. Currently, many researchers perform normal or abnormal classification by extracting characters or numeric features from traffic packets.

Fahad *et al.* [13] proposed a Global Optimization Approach (GOA) and used feature selection methods to classify spatial and temporal domain traffic data to 97% accuracy. Bang *et al.* extracted the temporal and spatial features of traffic data from LTE signaling and used the semi-Markov model to detect attacks in wireless sensor networks. Their method can effectively separate attack nodes and the false positive rate is very low [14]. Yang [15] proposed a new type of abnormal network traffic detection algorithm in the cloud computing environment. They proposed an Ent-SVM abnormal traffic detection system framework mainly considering the source IP address number, source port number, destination IP address number, destination port number, packet type number and network packet number. By calculating

the mixed information entropy and the eigenvalues of the canonical network, the SVM algorithm is used for intrusion detection. The proposed model can detect network abnormal traffic with high precision on large-scale datasets. Ertam and Avcı [16] proposed a GA-WK-EML network traffic classification model. They use genetic algorithms to select the best parameters based on the Wavelet function algorithm Extreme Learning Machine (WK-ELM). Through the adjustment of parameters, the accuracy of traffic classification exceeds 95%. Nezhad *et al.* [17] extracted the number of packets and the number of source IP addresses from the network traffic as the traffic detection indicator per minute to detect DoS and DDoS attacks. They built a time series of packet numbers and normalized them using the Box-Cox transformation. The ARIMA model is proposed to predict the number of packets every other minute, and then the chaotic behaviors of the prediction error time series are detected by calculating the maximum Lyapunov exponent. Through simulation, it is found that the number of data packets and the number of source IP addresses increase sharply during the attack time, and the classification accuracy rate for normal and attack traffic reaches 99.5%. Li *et al.* [18] proposed a multi-layer anomaly traffic detection model, which extracts the features of different network layers and uses PCA and random forest algorithms to remove redundant features. The detection accuracy and false positive rate of the model are improved by obtaining high-quality features. Roy *et al.* [19] designed a response feature from the KDD Cup99 dataset and classified the traffic using a deep neural network. The experimental results show that the deep neural network has better classification accuracy than SVM. Zhou *et al.* [20] extracted 256 features from the flow and mapped them into 16*16 grayscale images, and then used the improved convolutional neural network to classify flow. Their model has a good classification result for data types with large data volume, but the classification of data types with small data volume is very poor. Yuan *et al.* [21] proposed a recurrent neural network model for deep learning. They extracted 20 fields from continuous flow packets sequence and generated a three-dimensional feature map using a sliding time window of length T. The experiment found that the proposed model reduced the error rate by about 5 percentage points compared to the traditional machine learning algorithm. Kim *et al.* [22] used the LSTM network to perform five classifications in the KDD dataset. Although the classification results are ideal, the KDD dataset is too old and there are only four types of attacks. These types of attacks are no longer sufficient for today's network intrusion detection research. However, we found that the previously mentioned methods use different flow features, and the datasets used have been released for a long time without including some recent new attack types. In addition, most researchers use a shallow classification model, which can achieve better classification results when the feature dimension is small, but when the amount of data used is large and the feature dimension is large, the classification effect will be poor.

In this paper, we do not artificially design and extract the characters or statistical features in the flow, but extract the original hexadecimal codes in the flow, by mapping the original codes into equal-length decimal numbers as the features of the flow. We designed an improved deep hierarchical network model to classify flow, the CICIDS2017 dataset and CTU dataset were used in the experiments. These two new datasets contain a large number of traffic packets and attack types. The experimental results show that the proposed model can still achieve 99.9% classification accuracy under the condition of more types and numbers of traffic. In the experimental section, we compared the existing methods of Wang *et al.* [23] in detail, and found that our model had fewer parameters and a very a low miss detection rate, and proved that our model can rapidly converge through experiments. The differences between our proposed solution and existing methods such as BWManager [24] and LTE signaling attack detection scheme [14] include: 1) We use deep learning methods rather than traditional machine learning algorithms or statistical learning methods. 2) Our method requires the use of original traffic data generated by network users for analysis and detection, rather than analyzing the resources of the communication system for attack detection. 3) Our approach can not only detect network attacks in specific networks such as SDN, but also detect most common attacks on the Internet and only require traffic data generated by these networks. Therefore, our method can detect a large number of attack types, but more importantly, it can satisfy the attack detection in the big data environment by using deep learning.

III. METHODOLOGY

In this section, we designed an anomaly traffic detection model named deep hierarchical network. The deep hierarchical network consists of two layers of the neural network algorithms model. The first layer is based on the improved LetNet-5 convolutional neural network to extract the spatial features of the flow, and the second layer uses the LSTM network to extract the temporal features of the flow. The two networks are simultaneously trained by cascading into a hybrid network to enable the network to automatically extract the spatial and temporal features of the flow. Before introducing the deep hierarchical network, we will first introduce the composition of the traffic data used by the training model.

A. DATA PREPROCESSING

In this paper, the original traffic packets are used as the network intrusion detection analysis. Compared with the commonly used artificial traffic packets data extraction method, the method we proposed can retain all the feature information of each traffic packet. We do not need to filter or design the traffic features that need to be extracted. In the Wireshark we can see that the original traffic packets are some hexadecimal codes, as shown in FIGURE 1.

The process of extracting traffic features is as follows:

(1) data: Each traffic packet has an Ethernet layer, a network layer, a transport layer, and an application layer. In this

Algorithm 1 Original Flow Data Extraction

Input: network traffic pcap files.
Output: original flow data and its labels.
Step 1: transform pcap files to txt files.
for each pcap **do**
  Get flows based on the five-tuple information of traffic packages.
  for each flow **do**
    Transform flow pcap file into txt file with tshark to get flow's original hexadecimal data
  end
end
Step 2: extract the original flow features from the txt files
create a null list as all of the flows feature vectors,**flows**=[].
for each txt file **do**
  create a null list as all of the packages feature vector in the flow,**flow_feature**=[].
  for each package **do**
    create a null list as each package feature vector in the flow,**pkt_feature**=[].
    get the **16th** to **175th** hexadecimal bytes of the package to generate the package feature vector.
    if the number of bytes in the package less than 176 **do**
      fill with 0 to the 175th byte
    if the number of package less than 10 in the flow **do**
      fill with 0 to the 10th package
    updatae each package feature vector, **pkt_feature**.
    add **pkt_feature** vector into the **flow_feature** vector
  end
  label the flow base on the five-tuple information in the last dimension of the flow_feature vector
  add flow_feature vector into the flow vector
end

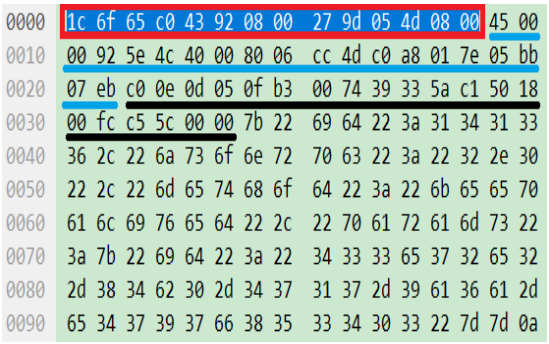


FIGURE 1. Raw traffic packet data.

paper, we do not use the data of the Ethernet layer and the network layer's Version and Differentiated Services fields. Because in the Ethernet layer, the three fields are the MAC source address, the MAC destination address, and the protocol version. According to Anderson *et al.*'s [12] analysis of the features of the flow, these fields are usually not used as the features of the traffic packets. The first line in FIGURE 1 is the raw data of a traffic packet we discarded.

(2) split: We use the SplitCap tool to split traffic packets with the same five-tupple information into a flow [25]. In the obtained flows, we found that the number of traffic packets contained in different flows is not the same within a certain timestamp. So we don't use all the traffic packets in a flow.

(3) vectorization: Statistics show that the number of traffic packets in most flows is less than 10, but the number of traffic packets in some flows is greater than 10 or even exceed 100. Since the payload length of each traffic packet is not equal, in order to use our raw data to train our classification model, we only extract 160 bytes in each traffic packet as the traffic packet features. Therefore, if the packet length of a packet is less than 160 bytes, then we need to use 0 padding for this packet. If a packet is longer than 160 bytes, we only take the first 160 bytes. In order to make the data sent to the model has the same dimensions, we only use the first 10 traffic packets of each flow. So, for each flow we extracted 1600-dimensional raw data. Original flow data extraction method is shown in Algorithm 1.

B. CNN MODEL

CNN's convolution operation has good spatial sensing ability, and it is widely used in image processing such as face recognition [26] and has achieved good results. In the network, the traffic packets generated by users are fragmented during the transmission process [27], and the IP field of each traffic packet indicates the spatial features of the flows. Considering the spatial features of traffic data, the first layer of our deep hierarchical network uses the CNN model to extract spatial features of traffic packets.

This paper uses the improved LeNet-5 network structure [28], which is a classic handwritten digit recognition

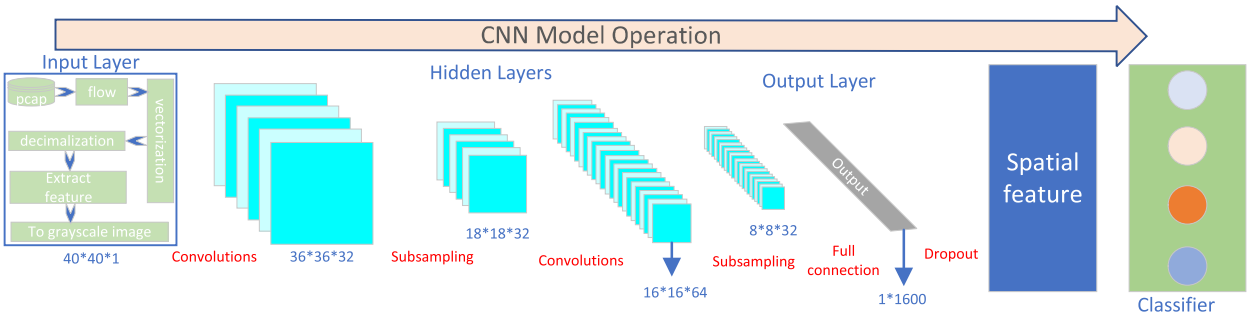


FIGURE 2. CNN network structure model

FIGURE 2. CNN network structure model.

CNN network. In this paper, the 1600-dimensional features are first converted into a 40*40 grayscale image as the input to the CNN network input layer. The hidden layer of CNN uses two convolution layers and two maximum pooling layers to perform spatial feature extraction on the original flow data. Among them, the first convolution layer uses 32 5*5 convolution kernels, and then performs the maximum pooling operation. The second convolution layer uses 64 3*3 convolution kernels and then performs maximum pooling operations. After convolution operations, the CNN hidden layer first uses the ReLU activation function to transform and then uses the maximum pooling operation. The original 40*40 grayscale image becomes 8*8 with 64 channels of an image. After performing a flatten operation on an 8*8*64 image, a 4096-dimensional vector is obtained and then sent to the output layer of the CNN. CNN's output layer uses a fully connected layer, and the fully connected layer uses 1600 neurons. The purpose is to maintain the same dimensional data feature as the original traffic data after spatial feature extraction. In addition, in order to prevent over-fitting, a dropout operation is performed after the fully connected layer to randomly inactivate some of the neurons of the fully connected layer. The CNN network structure used in this paper is shown in FIGURE 2.

The convolution operation uses an f*f-sized convolution kernel ω to perform a sliding convolution on a size n*n picture, and each sliding convolution produces a new feature. Suppose X is the input of the convolution, b is the bias term, c_i is the new feature produced by the convolution at the i-th layer, and σ_r is the activation function ReLU. Then the new features obtained by the convolution operation are:

$$c_i = \sigma_r\left(\omega * X_i + b_i\right) \tag{1}$$

After the convolution operation, the feature map of $n * n$will generate a feature map of $c = (n-f+1) * (n-f+1)$ size through a convolution kernel sliding window of size $f * f$. Maximum pooling is carried out for feature map c after convolution, and the maximum value in the selected window is taken as the final feature. The final feature map size is: $[(n-f+1) * (n-f+1)]/2$.

C. LSTM MODEL

The Recurrent Neural Network (RNN) in deep learning is widely used in speech processing, and has achieved good results in speech recognition and time series processing. In the traffic data, the transmission of the traffic packets has a chronological order, and the arrival of the traffic packets also has a sequence in the receiving end due to the delay problem. At the same time, the number of traffic packets sent within a certain timestamp varies, and these traffic packets characteristics indicate that they have temporal features. This paper uses the LSTM [29] network structure, and the LSTM network structure is a variant of RNN. The cell processor structure in the LSTM algorithm determines whether or not to add a useful message. Since the cell contains data operations for the input gate, the forget gate, and the output gate control network, this has a good effect in dealing with the dependency problem of a long sequence. The cell structure is shown in FIGURE 3.

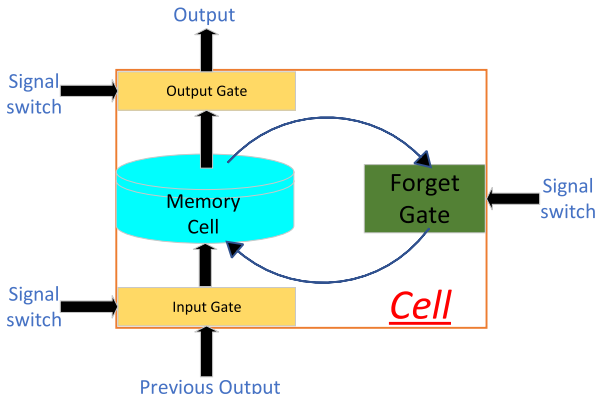


FIGURE 3. LSTM cell.

The calculation operation of each neuron node in LSTM is as follows:

(1)forget gate: The first step in the LSTM network is to determine the information to be discarded from the cell, which is done through the forget gate layer. The forget gate first reads the data information of the previous hidden layer

h_{t-1} and the input layer x_t, and then outputs a value between 0 and 1 to the cell state C_{t-1} through the activation function. 1 indicates complete reservation of information, and 0 indicates complete discard of information.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{2}$$

(2)input gate: The input gate determines how much new information is stored in the cell state. The update of the cell state consists of two steps: first, the input gate layer (sigmoid layer) determines the value to be updated by the cell, and then the tanh layer creates a candidate value vector C to added it the cell state.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{3}$$
$$\tilde{C}_t = \tanh(W_C.[h_{t-1}, x_t] + b_C) \tag{4}$$
$$C_t = f_t * \tilde{C}_t - 1 + i_{t-1} * \tilde{C}_t \tag{5}$$

(3)output gate: In order to determine the final output value, it is necessary to determine the state of the cell. First, use the sigmoid layer to determine which parts of the cell state to output. Then, the cell output is multiplied by the output of the previous sigmoid layer by a tanh layer operation as the final output value. The purpose of the tanh layer is to map cell state values between -1 and 1.

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{6}$$
$$h_t = o_t * \tanh(C_t) \tag{7}$$

Since the arrival time of the traffic packets in each flow is different and the values of the fields such as TTL are also different. Different from the methods of dealing with temporal feature like Feghhi and Leith [30] and Shen *et al.* [31], this paper uses the LSTM network to perform automatic temporal feature extraction on the original flow data. In this paper, the LSTM network uses two layers of cells for temporal feature extraction. Each cell of LSTM uses 256 hidden layer units. The cell activation function of each layer uses the sigmoid function for nonlinear operation. The last layer of the LSTM network uses a fully connected layer, and the number of neurons in the fully connected layer is equal to the number of classes of flow.

D. DEEP HIERARCHICAL NETWORK

Ahuja [27] showed that network flows contain a large number of features that can be analyzed. However, these features are based on statistics. These features, which are designed by hand, cannot express the temporal and spatial characteristics of flows by using traditional algorithms. These artificially designed features transform the intrinsic features of flows from the very beginning, and also lose some of the features of flows, so the high-level semantics of flows cannot be fully represented. The CNN and LSTM networks, along with deeper depths and using the original flows data can learn a high degree of semantic features and improve the performance of all metrics.

Since CNN and LSTM network can only extract the spatial feature and temporal feature of flows separately and can not fully express all the feature information of traffic, this paper can extract the spatial feature and temporal feature of flow simultaneously by forming a hybrid network structure model by combining CNN and LSTM networks. The hybrid network structure model is divided into two parts. Since the inputs to the CNN and LSTM network structures have different forms, we reshape the spatial features of the CNN network output at the junction of the CNN and the LSTM network. Since each flow extracts the first 10 traffic packets, each traffic packet extracts only the first 160 bytes. To correspond to each traffic packet, we make the input size of the LSTM network 160 and the input time step to 10. The output of the deep hierarchical network model is the probability of belonging to a certain kind of flow, and its structure is shown in FIGURE 4.

In this paper, the deep hierarchical network model structure classifier uses the softmax classifier, and the softmax classifier outputs the class probability of each type of flow. The index with the highest probability is the classification result of the hierarchical network on a flow. The loss function used in the model is the mean square loss function, and the training optimizer uses AdamOptimizer [32],which uses adaptive moment estimation for gradient descent. The training and testing process of the deep hierarchical network structure model is shown in Algorithm 2.

IV. EXPERIMENTS

In this section, we performed three different experiments on the CICIDS2017 dataset and the CTU dataset respectively. In the first experiment, we used the CNN model to extract the spatial features of the flow to classify it. In the second experiment, we used the LSTM model to extract the temporal features of the flow to classify it. In the third experiment, we extracted the spatial and temporal features of flow using the proposed deep hierarchical network model to classify it. Our experimental environment is as follows:

CPU: Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
GPU: GTX1080ti 11GB
RAM: 32GB
OS: Ubuntu 16.04

A. DATASET

As described by Weller-Fahy *et al.* [1], A key issue with most intrusion detection datasets is the lack of a sufficient number and types of traffic packets. This article uses two different datasets to conduct experiments, both of which were recently released and these two dataset contain more traffic and types. Reliable validation and test dataset compared to other datasets.

(1)CICIDS2017 Dataset

The CICIDS2017 dataset is an intrusion detection and intrusion prevention dataset that was open sourced by Canadian Institute for Cybersecurity in 2017. Sharafaldin *et al.* [33] designed a real attack scenario to collect traffic data by designing an attack network and a victim network. This dataset collects benign traffic and the most common attack traffic from Monday to Friday and

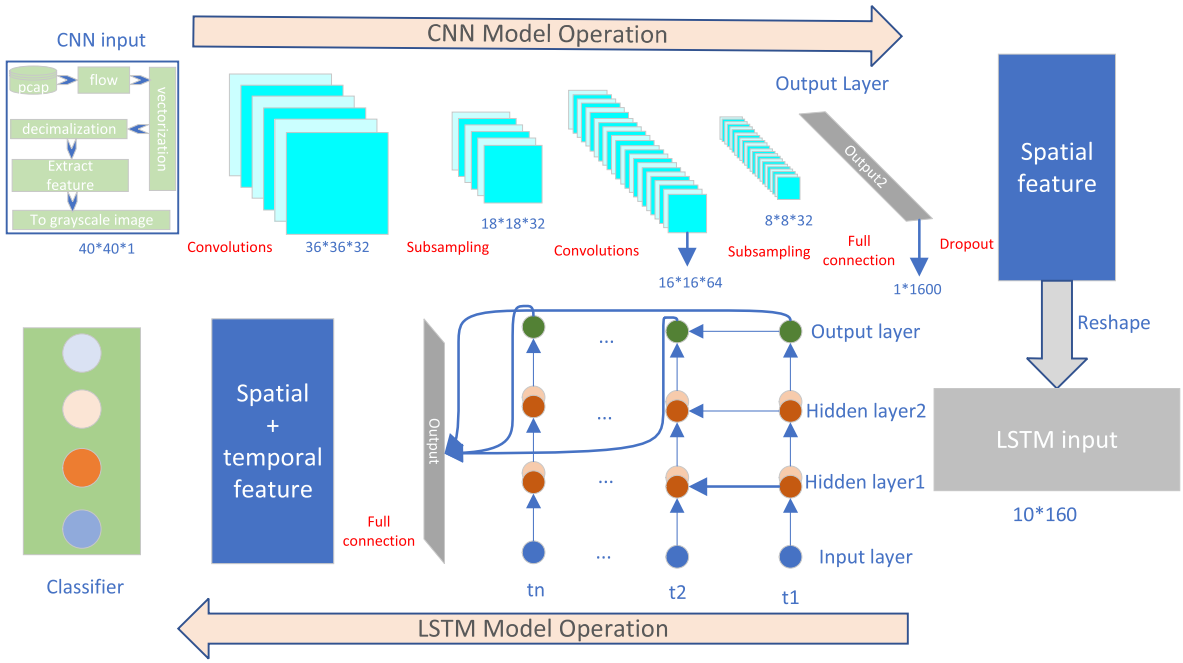


FIGURE 4. Deep hierarchical network structure model.

Algorithm 2 Training and Testing Process of the Deep Hierarchical Network

Input: Network flow images, each flow image include 10 packages(p1,p2…p10).
Output: flow category probabilities list [c1,c2…cn].
Step 1: CNN model learn spatial features
1. Reshape the 1600-dimensional flow feature into a 40*40 grayscale image.
2. Add the first layer of convolution operation(filter size:5*5*32)followed by the first max pooling layer of size 2*2.
3. Add the second layer of convolution operation (filter sizeÅĄÅ§3*3*64) followed by the second max pooling layer of size 2*2.
4. Add a full connection layer with 1600 neurons and then perform a dropout operation to obtain 1600-dimension temporal features.

Step 2: LSTM model learn temporal features
1. Reshape the temporal features into a 10*160 feature map.
2. Add the first lstm cell with 256 neurons.
3. Add the second lstm cell with 256 neurons.
4. Add a dense layer whose output are spatial&temporal features.
5. Add a softmax layer to output the probability of each class of flow.

Step 3: Train hybrid model
1. Add a mean square error loss function.
2. While train iteration do not complete,do
   a. Get a mini-batch train dataset as the model input.
   b. Compute the mean square erroe loss function j=1,n is the number of flow classes.
   c. Update the weights and biases using the Aamoptimizer gradient descent optimization algorithm.

end

Step 4: Test model
1. Test the trained model using the test dataset
2. Return the class of each test flow.

gives real-world pcap files data. On Monday, no attack traffic collected only benign traffic and the attack network launched an attack on the victim network to collected traffic generated by the network for a fixed period of time every Tuesday through Friday. Finally, the author accurately labeled the flow according to the timestamp of the flow, the source IP, the

destination IP, the source port, the destination port, and the protocol.

This paper extracts the benign flow and 10 types of attack flow from the CICIDS2017 dataset as the training and test data of the deep hierarchical network model, and extracts the flow features by extracting the original flow data in section 3. We labeled our generated flow according to the CICIDS2017 data labeling method to get a real and reliable label. Finally, we extract the number and type distribution of flows as shown in TABLE 1.

TABLE 1. CICIDS2017 dataset.

Flow type		Number	Percentage
Benign		339621	61.32%
DoS GoldenEye		7458	1.35%
DoS Hulk		14108	2.55%
DoS Slowhttp		4216	0.76%
DoS Slowloris		3869	0.70%
SSH Patator		2511	0.45%
FTP Patator		3907	0.71%
Web Attack	Brute Force	1353	0.24%
	SQL Injection	12	0.00%
	XSS	631	0.114
BotNet		1441	0.26%
Port Scan		158673	28.65%
DDoS		16050	2.90%

According to the number of flows we extracted in TABLE 1, it can be found that the percentage of benign flow and port scan attack flow are far greater than other types of the attack flow. In the multi-classification training, in order to deal with the deviation caused by data imbalance, we perform random downsampling on benign flows and port scan flows. In the binary classification, we use all the benign flows and attack flows.

(2)CTU Dataset

The CTU dataset is the BotNet traffic data collected by CTU University. This dataset contains a large amount of BotNet traffic and is mixed with normal traffic and background traffic. This dataset takes into account different types of BotNet traffic in different scenarios. The traffic in the PCAP files format is captured in each scenario and the traffics are labeled. This paper selects 11 types of traffic generated between April 2017 and April 2018. It includes 1 type of benign traffic and 10 types of BotNet traffic. The specific quantities of various types of traffic are shown in TABLE 2.

Since the percentage of benign flow and attack flow in the CTU dataset is not very different, we do not need to adopt data balance processing when performing multi-classification and binary classification. Although the Percentage of the total number of BotNet traffic between Viaxmr and Trojan is relatively small, considering that these two type flows are the more common attack flow, we still use them for intrusion detection analysis to find suspicious attack behaviors.

B. IMPLEMENTATION DETAILS

We train the hierarchical network in a joint end-to-end training method. The input to the CNN network is a 40*40 grayscale image, the first layer convolution operation is 32 5*5 kernels, and the second layer convolution operation is 64 3*3 kernels. The final 40*40 grayscale image is downsampled as 8*8*64, and a 1600-dimensional feature is output through a fully connected layer. Because the LSTM network inputs a time signal once within a timestep, we divide the 1600-dimensional feature of the CNN network output into a matrix size of 10*160. The reason for dividing into 10 inputs is because the 1600-dimensional feature represents 10 consecutive traffic packets in a flow, thus preserving the temporal features of the flow. The LSTM network consists of two layers, each layer with 256 neurons. We trained 1 epoch on the training set, and the mini-batchsize was 128. In the test phase we used mini-batchsize 2000.

TABLE 2. CTU dataset.

Flow type	Number	Percentage
Benign	64393	29.08%
Sathurbot	15659	7.07%
Trickster	14454	6.53%
TrickBot	61865	27.94%
Dridex	6805	3.07%
WebCompanion	11552	5.22%
Viaxmr	1028	0.46%
Trojan	1021	0.46%
CoinMiner	18684	8.44%
HTBot	15385	6.95%
Ursnif	10581	4.78%

The training method uses joint end-to-end training to train both CNN and LSTM networks. The forward process trains the CNN network and then trains the LSTM network. The backward process first calculates the loss of the LSTM and then calculates the loss of the CNN network. The joint end-to-end training method can ensure that the hierarchical network can simultaneously learn the temporal features and spatial features of flows and improve the classification accuracy and other metrics in the test phase.

C. EVALUATION METRICS

Our evaluation of model performance is based on the following metrics:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{8}$$

$$\Pr ecision = \frac{TP}{TP + FP} \tag{9}$$

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

$$F1 - Measure = \frac{2 * Precision * Recall}{\Pr ecision + Recall} \tag{11}$$

Here, TP is the number of positive samples in the test dataset and the model classification is also classified as positive samples. FP is the number of samples that are actually negative samples in the test dataset but are classified as positive samples. TN is the number of negative samples actually measured in the test dataset and the model is also classified as negative samples. FN is the number of test samples that

are actually positive samples but the model is classified as negative samples.

D. RESULTS

According to the original features we extracted from the flow, we perform binary classification and multi-classification on the CNN model, the LSTM model and the deep hierarchical network model respectively. The binary classification experiment performs normal and abnormal classification on flows, and the multi-classifications experiment performs a class of normal and ten kinds of abnormal classification on flows. The experimental results on the CICIDS2017 dataset are shown in TABLE 3, and the experimental results on the CTU dataset are shown in TABLE 4. CNN2 indicates that the binary classification is performed on the CNN algorithm, CNN11, which indicates that the CNN algorithm performs multi-classifications (1 benign plus 10 abnormal flows), and CNN+LSTM2 indicates that the binary classification is performed on our proposed deep hierarchical network. LSTM2, LSTM11 and CNN+LSTM are the same.

From the experimental results on the CICIDS2017 dataset in TABLE 3, we can find that the deep hierarchical network model proposed by us has better performance than the traditional machine learning algorithm model by Sharafaldin *et al.* [33]. They manually extracted 80-dimensional features from each flow for learning. Our model has better experimental (improved the classification accuracy by about 3%) results on the three metrics of precision, recall and F1-measure. At the same time, the accuracy metric we have given shows that our proposed deep hierarchical network model has a good detection efficiency for abnormal traffic. Although the proposed hierarchical network model has only a slight performance improvement compared with the CNN or LSTM model alone, in the actual network environment, because the amount of traffic data is very large, it is better to detect the traffic packets with attack behaviors as many as possible.

On the CTU dataset, the experimental results of the deep network model we proposed are shown in Table 4, compared with the experimental results of Huang *et al.* [34].

TABLE 3. CIC2017 dataset experiment results.

Metrics	Accuracy	Precision	Recall	F1-measure
Proposed Methods				
CNN2	0.998545	0.997576	0.998524	0.99805
LSTM2	0.99417	0.994401	0.990289	0.992341
CNN+LSTM2	**0.999125**	**0.998499**	**0.999251**	**0.998875**
CNN11	0.997778	**0.999419**	0.999543	**0.999481**
LSTM11	0.992722	0.997202	0.99716	0.997181
CNN+LSTM11	**0.998111**	0.998475	**0.999847**	0.999161
Sharafaldin's Method[27]				
KNN	-	0.96	0.96	0.96
RF	-	0.98	0.97	0.97
ID3	-	0.98	0.98	0.98
Adaboost	-	0.77	0.84	0.77
MLP	-	0.77	0.83	0.76
Naive-Bayes	-	0.88	0.04	0.04
QDA	-	0.97	0.88	0.92

TABLE 4. CTU dataset experiment results.

Metrics	Accuracy	Precision	Recall	F1-Measure
Proposed Methods				
CNN2	0.998174	0.998522	**0.998898**	0.99871
LSTM2	0.994261	0.998398	0.993067	0.995725
CNN+LSTM2	**0.998217**	**0.99912**	0.998382	**0.998751**
CNN11	0.970087	0.998363	0.992131	0.995237
LSTM11	0.963913	0.997494	0.99155	0.994513
CNN+LSTM11	**0.987696**	**0.99843**	**0.998082**	**0.998256**
He Huang's Methods[30]				
HDSN	-	94.7	94.9	-
STL	-	97.7	96.3	-
MIT	-	97.9	98	-

TABLE 5. Influence of input data size on classification accuracy.

Model	CNN	LSTM	CNN+LSTM
Header&Payload	**99.57146889**	**99.39217945**	**99.63495497**
Packet header	96.95600128	92.49089813	97.21829987
Payload	40.61399841	40.61399841	40.61399841

He Huang's method only gives two metrics of precision and recall, and we give the four metrics of accuracy, precision, recall and F1-measure. The experimental results show that our model is better on the two metrics of precision and recall. We retained more accurate values to compare performance between models more efficiently.

The three different experiments of CNN model, LSTM model and deep hierarchical network model on two datasets show that CNN network model and LSTM network model can extract spatial features and temporal features of flow separately. The separate CNN model and LSTM model can achieve good classification results in the binary-classification and multi-classification experiments. But comparing our proposed deep hierarchical network model to extract the spatial and temporal features of flow at the same time, our model can further improve the performance of these classification metrics. This shows that our proposed deep hierarchical network model can indeed learn the deeper abstract features of flow and perform better. The experimental results on both datasets are very good, indicating that our model has good generalization ability.

In order to study the influences of input data size and type on experimental results, we further studied the impacts of individual header data and payload on classification accuracy. Specifically, for each flow we extract the header and payload of the first five traffic packets. For each traffic packet, we extract the first 50 bytes of the header and payload respectively. By padding, we extend a flow to a 256-dimensional feature vector and then reshape the network to a size of 16*16. We used the header and payload raw data to conduct multi-classification experiments on the models we designed. The experimental results are shown in TABLE 5.

Through experimental result in TABLE 5, we find that the packet header information has more classification capability than the payload information. In particular, when the payload information is used alone, the model does not have the ability

TABLE 6. CICIDS2017 dataset semi-supervised model experimental results.

Clusters	2	**3**	4	5	6	7	8	9	10
Accuracy	94.8701	**94.9396**	94.8744	94.8605	94.7814	94.857	94.9014	94.8448	94.8396

TABLE 7. CTU dataset semi-supervised model experimental results.

CLusters	2	3	**4**	5	6	7	8	9	10
Accuracy	84.2607	84.1506	**84.4358**	84.3133	84.3612	84.2683	84.2817	84.3353	84.2923

to recognize when performing multi-classification. This is because in most cases the differences between payloads transmitted by the same host are not obvious and the payloads of the transmission are few, resulting in a very sparse feature matrix. Compared with the proposed method, by extracting the first 160 bytes of each traffic packet that include the packet header information and the payload information, the classification accuracy can be further improved under the same network structure. The gain obtained by the combined packet header and payload is mainly due to the addition of field information of the application layer, which further enhances the expression features of the traffic to make the traffic data more distinguishable. In fact, by analyzing the payload part, we found that the obtained feature vectors are very sparse and have too many 0 elements, which make our network models unable to distinguish the categories well.

In addition, we used the statistical features of Vlăduţu *et al.* [9] and the semi-supervised machine learning algorithm model of the Kmeans+Decision Tree on CICIDS2017 and CTU datasets for multi-classification experiments. The experimental results on the two datasets are shown in TABLE 6 and TABLE 7 respectively.

Through the experimental results in TABLE 6 and TABLE 7, we found that the classification accuracy of flow on the CICIDS2017 dataset exceeded 94%, but the experimental results were inferior to the experimental results of the deep hierarchical network model proposed by us. The experimental results on the CTU dataset are more than 10 percentage points worse than our proposed deep hierarchical network model. The experimental results show that the statistical features and traditional machine learning algorithms can not express the flow information as much as possible, which leads to the bottleneck of classification accuracy.

Further, we find that the network structure proposed by Wang *et al.* [23] is partially similar to the model proposed by us, but our model can perform better than their experimental results with fewer model parameters and converge very quickly. The difference in the recall metric is very obvious, we can reach 99.98% but they can only reach 96.91%, which indicates that our model has a very low miss detection rate. In fact, one-hot-encoding operation is adopted in the data preprocessing part of their model, which not only introduces feature engineering operation but also introduces a large number of useless parameters to increase the computational complexity of the model. Because the operation of their one-hot-encoding is to deal with each traffic packet, assuming a traffic packet of length n and an OHE vector length of m, then their method introduces $n\,(m-1)$ 0 elements. These large 0 elements account for $(1-1/m)$ percent of the total traffic packet bytes, which not only introduces additional computational parameters but also makes network learning useless. What's more, their hierarchical network structure is very different from ours. In their network structure, CNN extracts features by convolution operation just for each traffic packet. Firstly, spatial feature extraction is carried out for r traffic packets in a flow, then feature vectors of r traffic packets are cached, and finally each feature vector is sent to the LSTM network for temporal feature learning. In this way, their four-layer CNN outputs r one-dimensional vectors for one flow, while our two-layer CNN outputs 1 one-dimensional vector for the whole flow and then sends it to the LSTM network according to the time step. This further makes the model have fewer parameters and greatly reduces the cache storage space, which is an important reason for the fast convergence of our model. In order to illustrate the convergence performance of our model in detail, we give the training and test time of multi-classification of the model on CICIDS2017 dataset, and the results are shown in TABLE 8.

TABLE 8. Model convergence analysis.

Model	CNN2	LSTM2	CNN+LSTM2	CNN11	LSTM11	CNN+LSTM11
Train(s)	207.35	29.84	602.67	190.02	30.83	589.44
Test(s)	11.68	14.67	15.50	12.57	23.83	24.72

In TABLE 8, the parameters of the experiment were set to be the same as all the above experiments. Only 1 epoch was trained and test time analysis was performed on each model in 112,000 test samples. From the experimental results, we found that CNN was more time-consuming than LSTM in the training stage, while LSTM was more time-consuming than CNN in the test stage. This is because the convolutional neural network finally recovers to the original input data size after downsampling. Many of the 0 parameters in the middle have become non-zero parameters through learning, so they become denser through the fully connected layer. For the LSTM network, since the data is input according to the time step, the calculation time of the model cannot be significantly reduced even in the test stage. Experimental results show that the hierarchical network model proposed by us in the test stage only about 26% more time consume compares to a single CNN network and LSTM network, but does not require additional computing resources.

TABLE 9. Machine learning algorithms convergence analysis.

Model	KNN	NB	LR	RF	DT
Accuracy	0.93399	0.62482	0.97829	**0.99367**	0.99352
Train(s)	445.29	385.96	10316.78	31.75	84.06
Test(s)	29406.41	6.61	0.64	0.63	0.25

In order to compare with traditional machine learning algorithms, we trained five classifiers including KNN, Naive-Bayes(NB), Logistic Regression(LR), Random Forest(RF), Decision Tree(DT) on the CIC2017 dataset, and each classifier was multi-classified using original flow data. We give the accuracy of these algorithms, the training time, and the test time, the results are shown in Table 9. We find that Random Forest can achieve the highest classification accuracy, but this is still lower than the result of our proposed algorithm (0.998111). In terms of convergence, these five algorithms are quite different. The test time of KNN is about 8 hours, because the algorithm needs to calculate a large number of Euclidean distances. The test time of Random Forest and Decision Tree is low, because the depth parameter of the tree is set relatively small to prevent overfitting. It indicates that only a small number of features are required to recognize the abnormal traffic, and that these strongly separable features are derived from header fields (Table 5 shows that header is the main separable feature). In addition, because the payload of transmission in the dataset is very small, the feature matrix is very sparse, which is also the reason why the test time of Random Forest and Decision Tree algorithm is reduced. In the actual network environment, an attacker usually does not send a small amount of payload. In this case, the feature matrix will not become sparse and the test time will become longer.

V. IMPORTANT FEATURE ANALYSIS

In order to explore why our proposed deep hierarchical network model and flow classification method based on original flow data can achieve such high accuracy. We further analyze the features that are important for the flow classification in the experiments and give the actual meanings of these important features. In this section, we use three different important feature analysis methods, and weight the average of the analysis results of the three methods and finally give the top nine feature scores.

A. THREE METHODS

The principle of three different feature analysis methods is based on the analysis method of ensemble trees, which is weight-based, gain-based and cover-based. The three different methods are described below.

1) WEIGHT-BASED

The weight-based [35] method is currently the most commonly used method, which measures the importance of features by counting the number of times a feature is divided when constructing a subtree. If a feature is divided more times during the construction of the ensemble tree, then the more important this feature is.

2) GAIN-BASED

The gain-based method is a classical feature importance analysis method proposed by Breiman [36] in 1984. Gain is the contribution of loss or impurities to all the divisions of a feature. The gain calculation formula for feature A in a tree is: g(D,A) = H(D) - H(D|A). Where H(D) is the information entropy of feature set D in a given tree, and H(D|A) is the conditional entropy of feature set D given the condition of feature A. The larger the Gain, the more important the feature is.

3) COVER-BASED

The cover-based method is the relative amount of specified features observed in the tree. For example, suppose there are 100 observations, there are 3 trees and 4 different features in the ensemble tree, and assume that the node observations of feature 1 in the three trees are 10, 5 and 2, then the value of cover of feature 1 is 17. Similarly, the larger the cover value of a feature, the more important the feature is.

B. RESULTS

Three different feature importance analysis methods were used to analyze the original flow data extracted on the CICIDS2017 dataset. We performed binary classification and multi-classification experiments on 1600-dimensional features. The experimental results are shown in TABLE 10 and TABLE 11.

TABLE 10. Feature importance analysis: binary classification.

Weight-based									
F6	F198	F358	F320	F188	F31	F360	F1321	F193	F148
122	44	30	26	24	20	20	20	20	20
Gain-based									
F6	F198	F320	F358	F193	F389	F2	F188	F31	F360
120	50	30	26	24	22	22	20	20	20
Cover-based									
F6	F198	F8	F193	F320	F358	F389	F188	F360	F31
98	52	32	28	28	24	22	20	20	20

According to the experimental results, we found that in the binary classification and multi-classification the features of the three different feature importance analysis methods have some overlap. This suggests that these repeated features do have a large impact on the classification results. We compare the actual meaning of a TCP packet field to add the feature scores obtained by the three methods and give the actual meaning of these features in a TCP packet. The results of the analysis are shown in Figure 5. The actual meaning of each field is shown in TABLE 12.

According to the actual meaning of the field of a TCP packet, we find that for multi-classification, the impact of the TCP payload field on the flow classification is the most important, and the impact of fragment offset field on the binary classification is the most important. Combined with

TABLE 11. Feature importance analysis: multi-classification.

Weight-based									
F8	F687	F47	F6	F388	F45	F161	F41	F538	F1446
32	32	32	30	28	26	22	20	20	20
Gain-based									
F6	F687	F47	F366	F8	F45	F46	F320	F161	F527
52	38	37	32	30	27	25	23	22	20
Cover-based									
F6	F47	F687	F26	F1327	F45	F366	F48	F39	F166
61	40	31	27	26	24	24	23	20	20

TABLE 12. The actual meanings of the fields.

tp	TCP payload	ws	Window size
fo	Fragment offset	p	Protocol
up	Urgent pointer	tl	Total length
an	Acknowledge number	sn	Sequence number
f	Flags		

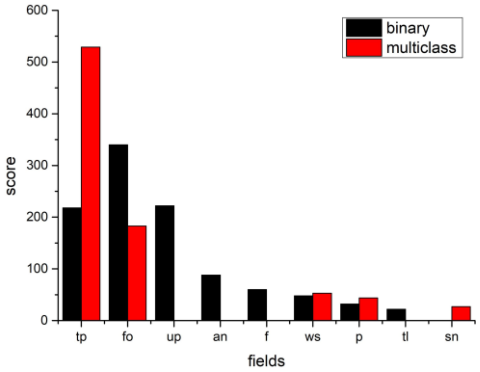


FIGURE 5. Important feature weighted score.

the urgent pointer, window size and acknowledge number fields, we can conclude that malicious flows are usually sent out in more slices. We found that several of the top 9 features are features that were previously rarely used by researchers in the field of network intrusion detection.

VI. CONCLUSION

We consider the artificial design and extraction of the features of the flow for network intrusion detection will lose part of the traffic information and thus affect the detection accuracy. In this paper, we extract the original information of flow and use our proposed hierarchical network to detect abnormal flow. Our hierarchical network is a specially designed CNN and LSTM model that learns spatial and temporal features from original flow information. To the best of our knowledge, this is the first time that the original information of flow is used for feature learning. The hierarchical network model we proposed is significantly better than other network intrusion detection models. In this paper, we use the CICIDS2017 dataset and CTU dataset. The experimental results on these two datasets show that our proposed model can achieve very high accuracy, precision, recall and F1-measure. At the same time, we analyzed the features that have contributed significantly to abnormal traffic detection and found features that were rarely used by previous researchers.

In the future work, we will design a traffic collection system by ourselves. Use our designed traffic acquisition system to collect real-world traffic data under the environment of our lab for analysis to detect suspicious attack traffic and evaluate test results. In addition, we will improve our hierarchical network model to make the network deeper, enabling the model to detect unknown types of attacks that have not been trained.

REFERENCES

[1] D. J. Weller-Fahy, B. J. Borghetti, and A. A. Sodemann, "A survey of distance and similarity measures used within network intrusion anomaly detection," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 70–91, Jan. 2015.

[2] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, Jan. 2016.

[3] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, "Internet traffic classification by aggregating correlated naive Bayes predictions," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 5–15, Jan. 2013.

[4] A. Dainotti, F. Gargiulo, L. I. Kuncheva, A. Pescapè, and C. Sansone, "Identification of traffic flows hiding behind TCP Port 80," in *Proc. IEEE Int. Conf. Commun.*, May 2010, pp. 1–6.

[5] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," in *Proc. 13th Int. Conf. World Wide Web*, pp. 512–521.

[6] K. Wang, G. Cretu, and S. J. Stolfo, "Anomalous payload-based worm detection and signature generation," in *Proc. Int. Workshop Recent Adv. Intrusion Detection*, 2005, pp. 227–246.

[7] R. T. El-Maghraby, N. M. A. Elazim, and A. M. Bahaa-Eldin, "A survey on deep packet inspection," in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 188–197.

[8] D. E. Denning, "An intrusion-detection model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[9] A. Vlăduţu, D. Comăneci, and C. Dobre, "Internet traffic classification based on flows' statistical properties with machine learning," *Int. J. Netw. Manage.*, vol. 27, no. 3, p. e1929, May 2017.

[10] M. Yeo *et al.*, "Flow-based malware detection using convolutional neural network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 910–913.

[11] Y. Yu, J. Long, and Z. Cai, "Session-based network intrusion detection using a deep learning architecture," in *Modeling Decisions for Artificial Intelligence*. V. Torra, Y. Narukawa, A. Honda, and S. Inoue, Eds. Cham, Switzerland: Springer, 2017, pp. 144–155.

[12] J. P. Anderson, "Computer security threat monitoring and surveillance," James P. Anderson Company, Washington, DC, USA, Tech. Rep. 19034, 1980.

[13] A. Fahad, Z. Tari, I. Khalil, A. Almalawi, and A. Y. Zomaya, "An optimal and stable feature selection approach for traffic classification based on multi-criterion fusion," *Future Gener. Comput. Syst.*, vol. 36, pp. 156–169, Jul. 2014.

[14] J.-H. Bang, Y.-J. Cho, and K. Kang, "Anomaly detection of network-initiated LTE signaling traffic in wireless sensor and actuator networks based on a hidden semi-Markov model," *Comput. Secur.*, vol. 65, pp. 108–120, Mar. 2017.

[15] C. Yang, "Anomaly network traffic detection algorithm based on information entropy measurement under the cloud computing environment," *Cluster Comput.*, vol. 21, pp. 1–9, Jan. 2018. [Online]. Available: https://link.springer.com/journal/volumesAndIssues/10586

[16] F. Ertam and E. Avcí, "A new approach for internet traffic classification: GA-WK-ELM," *Measurement*, vol. 95, pp. 135–142, Jan. 2017.

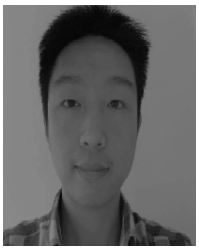
[17] S. M. T. Nezhad, M. Nazari, and E. A. Gharavol, "A Novel DoS and DDoS attacks detection algorithm using ARIMA time series model and chaotic system in computer networks," *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 700–703, Apr. 2016.

[18] B. Li, S. Zhang, and K. Li, "Towards a multi-layers anomaly detection framework for analyzing network traffic," *Concurrency Comput. Pract. Exper.*, vol. 29, no. 14, p. e3955, Jul. 2017.

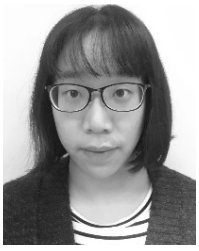
[19] S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. V. Krishna, "A deep learning based artificial neural network approach for intrusion detection," in *Mathematics and Computing*, D. Giri, R. N. Mohapatra, H. Begehr, and M. S. Obaidat, Eds. Singapore: Springer, 2017, pp. 44–53.
[20] H. Zhou, Y. Wang, X. Lei, and Y. Liu, "A method of improved CNN traffic classification," in *Proc. 13th Int. Conf. Comput. Intell. Secur. (CIS)*, Dec. 2017, pp. 177–181.
[21] X. Yuan, C. Li, and X. Li, "Deepdefense: Identifying DDoS attack via deep learning," in *Proc. IEEE Int. Conf. Smart Comput. (SMARTCOMP)*, May 2017, pp. 1–8.
[22] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Feb. 2016, pp. 1–5.
[23] W. Wang *et al.*, "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2018.
[24] T. Wang, Z. Guo, H. Chen, and W. Liu, "BWManager: Mitigating denial of service attacks in software-defined networks through bandwidth prediction," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 4, pp. 1235–1248, Dec. 2018.
[25] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, "Robust smartphone app identification via encrypted network traffic analysis," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 1, pp. 63–78, Jan. 2018.
[26] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Netw.*, vol. 16, nos. 5–6, pp. 555–559, Jul. 2003.
[27] R. K. Ahuja, *Network Flows: Theory, Algorithms, and Applications*. London, U.K.: Pearson Education, 2017.
[28] Y. LeCun *et al.* (2015). *Lenet-5, Convolutional Neural Networks*. [Online]. Available: http://yann. lecun. com/exdb/lenet
[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[30] S. Feghhi and D. J. Leith, "A Web traffic analysis attack using only timing information," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 8, pp. 1747–1759, Aug. 2016. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2016.2551203
[31] M. Shen, M. Wei, L. Zhu, and M. Wang, "Classification of encrypted traffic with second-order Markov chains and application attribute bigrams," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 8, pp. 1830–1843, Aug. 2017.
[32] D. Kinga and J. B. Adam, "A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, vol. 5, 2015, pp. 1–9.
[33] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, Jan. 2018, pp. 108–116.
[34] H. Huang, H. Deng, J. Chen, L. Han, and W. Wang, "Automatic multitask learning system for abnormal network traffic detection," *Int. J. Emerg. Technol. Learn.*, vol. 13, no. 4, pp. 4–20, Jan. 2018.
[35] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jun. 2016, pp. 785–794.
[36] L. Breiman, *Classification Regression Trees*. Evanston, IL, USA: Routledge, 2017.

YONG ZHANG received the Ph.D. degree from the Beijing University of Posts and Telecommunications, China, where he has been an Associate Professor with the School of Electronic Engineering. His research interests include self-organizing networks, mobile communications, and cognitive networks.

XU CHEN received the B.S. degree in communication engineering from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2017. He is currently pursuing the B.S. degree with the Beijing University of Posts and Telecommunications, Beijing, China. His research interests include deep learning and intrusion detection.



LEI JIN received the B.S. and M.S. degrees in electrical and electronics engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the School of Electronic Engineering. His current research interest includes artificial intelligence.



XIAOJUAN WANG received the Ph.D. degree in electrical and electronics engineering from the Beijing University of Posts and Telecommunications, Beijing, China, where she is currently an Associate Professor with the School of Electronic Engineering. Her current research interest includes artificial intelligence.

DA GUO received the Ph.D. degree in electrical engineering from the Beijing University of Posts and Telecommunications, where he is currently a Senior Engineer. His research interests include mobile communications, opportunistic networks, WSN, and P2P networks.