Computers, Materials & Continua CMC, vol.62, no.3, pp.1317-1333, 2020

DDoS Attack Detection via Multi-Scale Convolutional Neural Network

Jieren Cheng[1, 2], Yifu Liu[1, *], Xiangyan Tang[1], Victor S. Sheng[3], Mengyang Li[1] and Junqi Li[1]

Abstract: Distributed Denial-of-Service (DDoS) has caused great damage to the network in the big data environment. Existing methods are characterized by low computational efficiency, high false alarm rate and high false alarm rate. In this paper, we propose a DDoS attack detection method based on network flow grayscale matrix feature via multi-scale convolutional neural network (CNN). According to the different characteristics of the attack flow and the normal flow in the IP protocol, the seven-tuple is defined to describe the network flow characteristics and converted into a grayscale feature by binary. Based on the network flow grayscale matrix feature (GMF), the convolution kernel of different spatial scales is used to improve the accuracy of feature segmentation, global features and local features of the network flow are extracted. A DDoS attack classifier based on multi-scale convolution neural network is constructed. Experiments show that compared with correlation methods, this method can improve the robustness of the classifier, reduce the false alarm rate and the missing alarm rate.

Keywords: DDoS attack detection, convolutional neural network, network flow feature extraction.

1 Introduction

Distributed denial of service (DDoS) attacks are currently used by hackers and they are difficult to guard against. It is an attack technology derived from Denial of Service (DOS) attack. This kind of attack initiated by an organized, distributed, or remotely controlled botnet [Aiko, Jose, Jessica et al. (2016)]. A DDoS attack combines multiple computer devices to send a large number of consecutive attacks [Behal and Kumar (2017)]. This attack attacks maliciously from multiple systems, making it impossible for computer or network resources to provide services to their established users. It is usually expressed as a service that interrupts or suspends connections to the Internet, thereby reducing the performance of the network. In this way, it can make the network paralyze [Yadav, Trivedi and Mehtre (2016)].

[1] School of Information Science and Technology, Hainan University, 570228, Haikou, China.

[2] State Key Laboratory of Marine Resource Utilization in South China Sea, 570228, Haikou, China.

[3] Department of Computer Science, University of Central Arkansas, Conway, AR 72035, USA.

[*] Corresponding Author: Yifu Liu. Email: yifu@hainanu.edu.cn.

2 Related work

According to the different algorithms, the detection of DDoS attacks can be divided into several methods. This paper mainly introduces the related research of detection based on statistical analysis method and machine learning method. Statistics-based DDoS attack detection by analyzing the regularity of eigenvalue statistics from a large amount of data. We use statistical methods to describe the changes in network traffic and packet structure caused by DDoS attacks. Based on the statistical method, quantitative analysis of aggressive behavior is carried out, statistical features are extracted and math model is used to detect aggressive behavior. This kind of method mainly includes entropy, information theory, statistical analysis, etc. Idhammad et al. [Idhammad, Afdel and Belouch (2018)] have proposed a DDoS attack detection method based on information theory entropy and random forest algorithm in cloud environment to estimate the entropy of network features of input network flow. Petkovic et al. [Petkovic, Basicevic, Kukolj et al. (2018)] have proposed a two-step method for detecting DDoS attacks in combination with network flow entropy and Takagi Sugeno Kang fuzzy system. Behal et al. [Behal and Kumar (2017)] have proposed a new information theory metric DDoS attack detection method based on phi-entropy and phi-diverging.

Machine learning-based DDoS attack detection by extracting feature sample sequences from the network flow. This kind of method uses machine learning algorithms to learn the training samples, then building a classifier to classify the test samples. The main methods include support vector machine (SVM), random forest (RF), naive bayes (NB), etc. Ye et al. [Ye, Cheng, Zhu et al. (2018)] have extracted 6-tuple characteristic values of the switch flow table, and then DDoS attack model is built by combining the SVM classification algorithms. Cheng et al. [Cheng, Yin, Liu et al. (2009)] have proposed a method for classifying normal network flow and attack network flow obtained by using support vector machine (SVM) classifier with IAI time series training. Wang et al. [Wang, Zheng and Li (2017)] have proposed a DDoS attack detection method based on RDF-SVM algorithm, which used random forest to calculate the importance of features, then used SVM to rescreen features to prevent false removal features. In recent years, researches have largely turned to hybrid technology detection methods to achieve better detection results. With the advent of the big data era, research on security analysis and strategy research under big data is increasing. Pandey et al. [Pandey, Peddoju and Deshpande (2017)] have proposed a statistical distributed network packet filtering model and optimization algorithm under cloud computing.

In recent years, with the rapid development of deep learning, it has also begun to be widely used in DDoS attack detection. Saied et al. [Saied, Overill and Radzik (2016)] have proposed an artificial neural network (ANN) based on specific features to detect DDOS attacks. This method separated the DDOS attack network flow from the real traffic. Tariq Ahanger [Ahanger (2017)] has proposed a DDoS detection method based on ANN, and this method trained the ANN model to detect normal and abnormal traffic by analyzing system resource and network data. Ham et al. [Ham and Kostanic (2000)] have proposed an anomaly detection algorithm based on neural network.

This paper proposes a detection model of convolutional neural network based on network flow grayscale matrix feature, and optimizes its parameters by using optimization

algorithm to detect DDoS attacks accurately and effectively. The attack characteristics and detection optimization of the algorithm are analyzed.

3 DDoS attack feature extraction

3.1 Analysis of DDoS attack characteristics

By studying the typical DDoS attack cases, the DDoS attacks have the characteristics that the attack sources are wide distributed and strong concealment of attack sources [Petkovic, Basicevic, Kukolj et al. (2018)], specific characteristics are as follows:

Wide distribution of attack sources. The source IP address and destination IP address of the attack have a "many-to-one" relationship. When a DDoS attack occurs, a large number of downtimes are controlled and simultaneously attack the specified target [Yuan, Li and Li (2017)]. The attacker can forge the source IP address of the attack packet continuously or randomly, making the distribution of source IP address decentralized more disperse [Yu, Hu and Wang (2018); Cheng, Xu, Tang et al. (2018)]. It makes the distribution of source IP address, source port and destination port number more dispersed. In the case of attack, most of the packets sent by the attacker are not segmented, and the number of packets with the IP flag of 0X4000 (Do not Fragment) will increase significantly.

Strong attack power. Because DDoS attacks use multiple attack sources to launch attacks at the same time, the traffic generated by each attack source is aggregated to form a huge attack traffic [Gao, Cheng, He et al. (2018)]. It breaks the upper limit of the processing power of the attacked target in a short period of time, causing the target system to fall into paralysis [Mamolar, Pervez, Calero et al. (2018); Cheng, Zhang, Tang et al. (2018); Mirkovic and Reiher (2004)].

For SYN Flood, the attacker will send a lot of SYN requests, and the server will consume a lot of resources to retry SYN+ACK. The attacker will send a lot of TCP flag bit 0X02 (SYN) packets. Due to the consumption of server resources, the returned TCP flag bit 0X10 (SYN+ACK) packets will gradually decrease with the increase (of)in attack intensity [Cheng, Liu and Tang (2018); Wang, Ma, Zhang et al. (2016)]. In the case of attack, the distribution state of TCP flags will change significantly.

This paper proposes grayscale matrix feature (GMF) by analyzing network traffic. Based on this feature, the DDoS convolutional neural network classifier is constructed, and the attack characteristics and detection performance of the algorithm are optimized.

3.2 Feature extraction rule

Given a network flow F with n sample IP packets, we define each IP packet as $(t_i, s_i, d_i, sp_i, dp_i, size_i, tf_i, if_i)$, t_i denotes the arrival time of the packet i, s_i and d_i denotes its source IP and the destination IP, sp_i denotes its source port and the destination port, $size_i$ denotes its packet size, tf_i and if_i denotes its TCP flags and IP flags respectively. Execute the following rules for these n packets:

(1) Binary conversion

In order to preserve each original attribute of the network flow F, we perform number conversion on the above s_i, d_i, sp_i, dp_i, $size_i$, tf_i, if_i. For the hexadecimal conversion, the bit-weight conversion method [Suzuki and Murayama (1985)] is used. Any hexadecimal data can be in the form of a sum of polynomials spread by bit weight. For example, the number N can be expressed by the following formula:

$$N=A_{n-1}\times R^{n-1}+A_{n-2}\times R^{n-2}+\ldots+A_1\times R^1+A_0\times R^0=\sum_{i=0}^{n-1}(A_i\times R^i) \quad (1)$$

According to formula 1, we convert s_i, d_i, sp_i, dp_i, $size_i$, tf_i, if_i to binary data respectively.

(2) Formal conversion

Among them, due to the problem that the number of bits in the network flow is inconsistent after being converted into binary, the digits are formally converted as following formula:

$$(data_i)_2=\begin{cases}Precomplement\ 0\ , & len[(data_i)_2]<L\\ (data_i)_2\ , & len[(data_i)_2]=L\end{cases} \quad (2)$$

L is the threshold, which is the length of $(data_i)_2$. $(data_i)_2$ represents the binary form of $data_i$, $len[(data_i)_2]$ represents the digits of binary data $data_i$. Because the original format of IP and port number is 32-bit binary, then set the threshold L=32. Converting the source IP address s_i, destination IP address d_i, source port sp_i, destination port dp_i of the packet of i to 32-bit binary data. Statistically, the length of the data package is less than 4096 bytes (2^{12}bytes), then set the threshold L=12, converting the packet size $size_i$ to 12-bit binary data. Because IP flags and TCP flags are all hexadecimal data in a given dataset, then set the threshold L=16, converting TCP flags tf_i, IP flags if_i to 16-bit binary data.

(3) Sampling by time

Converted by the above number system to obtain a binary form network flow $F'=<(t_i, s_i, d_i, sp_i, dp_i, size_i, tf_i, if_i)>$, $i=1, 2, \ldots, n$

Definition 1. Based on the binary representation, the network flow data is sampled by unit time Δt, Packet sampling time $T\in(0,N)$,Δt=0.01s, 0.05s, 0.1s, we extract the packet set(PS):

$$PS=\sum\{t_i, s_i, d_i, sp_i, dp_i, size_i, tf_i, if_i\}_{\Delta t} \quad (3)$$

In the definition of PS, in order to analyze the state characteristics of the PS more efficiently, statistics on the network flow $F'=(t_i, s_i, d_i, sp_i, dp_i, size_i, tf_i, if_i)$ per unit time are performed, we analyze the law of network traffic generation.

DDoS attack is a process in which an attacker uses a large number of forged source IP addresses to send useless packets to the victim host, consuming the target host resources and causing the attack. Therefore, when DDoS attacks occur, a large number of false source IP addresses will be generated per unit time, source IP addresses will increase, and destination IP addresses will be relatively single. The number of different destination port increases abnormally when useless packets are sent from the attack source IP to multiple target ports of the target host in a unit time.

In the process of TCP packet transmission, attackers will forge addresses to send SYN requests to servers, and a large number of packets with TCP flag bit 0X02 (SYN=1) will appear, at the same time, the server sends a packet with a flag of 0X10 (SYN, ACK=1) to the requester for confirmation. Because the requester is a forged address, the server will not receive a response. With the increase of attack traffic, the server will consume a lot of resources to handle this kind of semi-connection, which will eventually lead to server crash. Statistics show that, in the process of attack, the proportion of packets with the IP flag of 0X4000 (Don't Fragment) sent by the attacker will increase significantly.

We extract GMF feature based on the above rules, and the network traffic eigenvalue is obtained according to the corresponding sampling time Δt. Because convolutional neural networks require consistently the size of input data, we traverse the PS matrix and perform grayscale encoding processing on the network flow features according to the following formula:

$$Gray\ PIC = \begin{cases} black, & a_i=0 \\ white, & a_i=1 \end{cases} \tag{4}$$

a_i is the network flow eigenvalue component.

According to the above feature extraction rules, we extract GMF feature:

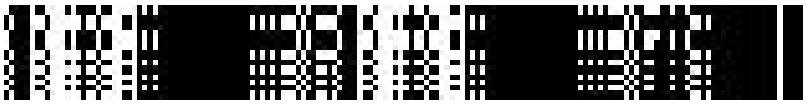

Figure 1: Grayscale matrix feature in normal flow

As shown in Fig. 1, we can see the network flow distribution of normal flow. The transverse axis of the matrix includes s_i, sp_i, d_i, dp_i, $size_i$, tf_i, if_i, the longitudinal axis of a matrix represents the number of packages. There are more "one-to-one" resource access modes. Normal flow rate is relatively low, and source IP addresses are relatively concentrated. The source port and destination port number are relatively concentrated. The value of TCP flags is relatively stable, and the IP flags are various, there are not only a certain number of segmented packages but also a certain number of unscheduled packages in the network flow.

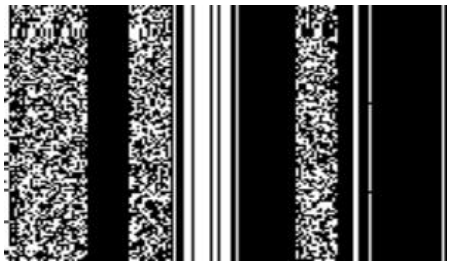

Figure 2: Grayscale matrix feature in DDoS flow

As shown in Fig. 2, we can see that the distribution of GMF feature changes in the case of DDoS attacks. There are many "many-to-one" resource access modes. Under the attack state, network flow embodies high flow characteristic, the number of data packets collected under the current sampling time is large. Source IP addresses and source ports are scattered,

destination IP addresses are centralized and destination ports are scattered. Significant changes in packet size distribution. The proportion of unbranched data packets increased significantly. TCP flags changed with the distribution of attack start state.

We can find the GMF features we extracted by binary conversion from original data. The time and space distribution of network flow attributes can be more accurately reflected by the representation of network flow related attributes in the form of matrix. They can more comprehensively express the spatial relationship and time-distance relationship of data packets.

The existing DDoS attack detection methods generally use statistical methods to extract network flow features. By analyzing the state changes of normal flow and attack flow the characteristic sequence of network flow is extracted by statistic the related attributes of network packets. The feature of network flow based on statistical often results in information loss to a certain extent in the statistical process. Statistical-based methods can't fully and accurately reflect the characteristics of network flow.

In summary, our proposed GMF features can more accurately reflect the distribution of data packets and the spatial relationship between data packets. Compared with statistical network flow feature sequences, GMF feature has stronger feature expression ability in spatial and temporal relationships.

4 Multi-scale convolutional neural network classifier

4.1 Matrix normalization

Since the sampled number of network flow features during the sampling time is different, we use the gray map mapping method to map the grayscale features:

$$f(x)=(width, height) \xrightarrow{\text{resize}} (W, H) \tag{5}$$

In the formula (5), width is the original width of the grayscale matrix, the weight is the original height of the grayscale image, W and H are threshold values. W is the threshold of width, H is the threshold of height. When the statistical sampling time is 0.01 s, according to statistics, the number of data packets collected does not exceed 300. When the statistical sampling time is 0.05 s, according to statistics, the number of data packets collected does not exceed 800. When the statistical sampling time is 0.1 s, according to statistics, the number of data packets collected does not exceed 1500. When sampling time is 0.01 s, set the width threshold W=172, the height threshold H=300. When sampling time is 0.05 s, set the width threshold W=172, the height threshold H=800. When sampling time is 0.1 s, set the width threshold W=172, the height threshold H=1500.

We obtain the grayscale network flow features. We divide the training set, verification set and detection set by the proportion of 0.8, 0.1 and 0.1. Training set is used for model fitting, validating set is used to adjust the hyper parameters of the model and preliminarily evaluate the capability of the model. Test set is used to evaluate the generalization ability of the model.

Based on the features extracted from the above methods, this paper uses CNN to build the detection model. It has become a research hotspot in the current image field. Its weight sharing network structure makes it more similar to biological neural network, which

reduces the complexity of the network model and the number of weights. This advantage is more obvious when the input of the network is multi-dimensional image and multi-dimensional matrix. The image and matrix can be directly used as the input of the network, avoiding the complicated feature extraction and data reconstruction process in the traditional recognition algorithm. Convolutional network is a multi-layer perceptron specially designed to recognize two-dimensional shapes. This network structure has some invariance to translation, scaling, and other forms of deformation. In a typical CNN, it generally represents the alternation of the convolutional layer and the pooling layer. The last few layers of the network near the output layer are usually fully connected networks. The training process of convolution neural network learns network parameters such as convolution kernel parameters and interconnection weights of convolution layer. The prediction process is mainly based on the input image and network parameters to calculate the category label.

The GMF feature proposed by us is the arrangement of high-dimensional matrices, because convolutional neural network has good performance for high-dimensional matrix processing. We use GMF features to train CNN model. Different convolution kernel sizes are determined according to the bit length of feature components in GMF feature. Because of IP address, port number is 32-bit binary, packet length is 12-bit binary data, due to the fact that TCP flag bit and IP flag bit are 16-bit binary data, we use convolution kernels of different scales for feature extraction. We put the matrices into the convolution layer:

$$x^l_j = f\left(u^l_j\right) \tag{6}$$

$$u^l_j = \sum_{i \in M_j} x^{l-1}_i * k^l_{ij} + b^l_j \tag{7}$$

In formula (6) and formula (7), u^l_j is the net activation of the j-th channel of convolution layer l, it is gained by convolution summation and offsetting the previous layer output feature map x_i^{l-1}, x_i^l is the output of the j-th channel of convolution layer l. $f(\cdot)$ is an activation function and uses functions such as sigmoid function and tanh function. M_j represents a subset of input feature maps used to calculate u^l_j, k^l_{ij} is the convolution kernel matrix, b^l_j is a bias to the convolution feature map. For an output feature map x^l_j, the convolution kernel k^l_{ij} corresponding to each input feature map x^{l-1}_j may be different. "*" is the symbol of convolution.

Then we put u^l_j into pooling layer:

$$x^l_j = f\left(u^l_j\right) \tag{8}$$

$$u^l_j = \beta^l_j down\left(x^{l-1}_j\right) + b^l_j \tag{9}$$

In formula (8) and formula (9), u^l_j is the net activation of the jth channel of the pooling layer l, It is obtained by pooling and offsetting the output feature map x_i^{l-1} of the previous layer, β is the weighting factor of the pooling layer, b^l_j is the offset of the pooling layer, $down(\cdot)$ is the pooling function. It divides the input feature map x^{l-1}_j into multiple non-overlapping $n \times n$ image blocks by sliding window method. The pixels in each image block are then summed, averaged or maximized, and the output image is then reduced by n times in both dimensions.

In a fully connected network, splicing the feature maps of all 2D images into one-dimensional features as input to a fully connected network, the output of the fully connected layer l can be obtained by weighting the inputs and obtaining the response through the activation function.

$$x^l = f\left(u^l\right) \tag{10}$$

$$u^l = w^l x^{l-1} + b^l \tag{11}$$

In formula (10) and formula (11), u^l is the net activation of the fully connected layer l, it is obtained by weighting and offsetting the output map x^{l-1} of the previous layer. w^l is the weight coefficient of the fully connected network, b^l is the offset of the fully connected layer l. The convolution model used in this paper includes two convolutional layers, two pooling layers, two local layers and a softmax layer to build our model.

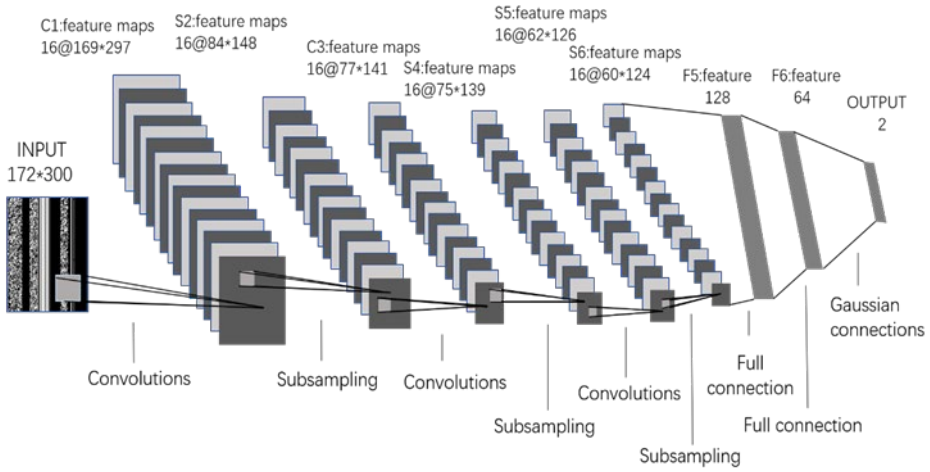

Figure 3: Construction of the multi scale GMF-CNN Model

As shown in Fig. 3, we optimized the convolution kernel size by mapping three 3*3 convolution kernels into 4*4, 8*8 and 16*16 multi-scale changes to construct the multi scale GMF-CNN model. The advantage is that, due to source IP, destination IP, source port number, and destination port number are 32-bit binary data. Due to the fact that we use 4*4, 8*8 convolution kernels adapt to the data format better. In addition, the 16*16 size convolution kernel can realize the dimensionality reduction of data, further adapt to the data form, and improve the detection capability of the model.

5 Experiment

5.1 Dataset and evaluation criteria

The experimental hardware devices in this article are 8G memory, i5 processor, and they are implemented in windows10 64bit system, Python 3.6.2 | Anaconda 4.2.0 (64-bit) environment.

The data set was experimented with the CAIDA "DDoS Attack 2007" data set, which contained approximately one hour of distributed denial of service (DDoS) anonymous traffic attacks on August 4, 2007. This type of attack attempts to block access to the

target server by consuming the computing resources on the server and all the bandwidth connected to the Internet. The total size of the data set is 2 GB, accounting for about one hour. The attack started at about 21:14, causing the network load to grow rapidly, from about 200 kilobits per second to 80 megabits per second. One hour of attack traffic is divided into 5 minutes of files and stored in PCAP format.

In order to reasonably judge the effectiveness of the proposed attack detection experiment, we use some evaluation indicators to fully explain its detection performance, including detection rate (DR), false alarm rate (FR), error rate (ER). Assuming that TP is the number of normal samples that are correctly marked, TN is the number of attack samples that are correctly marked, FN is the number of attack samples that are incorrectly marked, and FP is the number of normal samples that are incorrectly marked.

$$DR=\frac{TN}{TN+FN} \tag{12}$$

$$FR=\frac{FP}{TN+FP} \tag{13}$$

$$ER=\frac{FN+FP}{TP+FP+TN+FN} \tag{14}$$

The detection rate is the probability that the actual attack can be detected. The false alarm rate describes the proportion of samples that are judged to be aggressive in normal samples. The error rate is the probability that the user behavior is wrongly judged.

5.2 Comparison of experimental results

In order to verify the detection ability of our proposed GMF feature combined with the multi-scale convolutional neural network method, the following features and algorithm comparison experiments were carried out.

5.2.1 Comparison of features

(1) Detection rate comparison

We extract features according to the methods described in Cheng et al. [Cheng, Zhang, Tang et al. (2018)]. According to the feature extraction rules in this paper, we use FFV statistical feature values of one-dimensional features of quintuple features for experimental comparison.

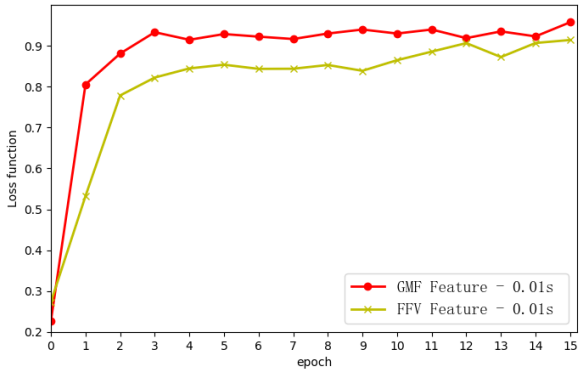

Figure 4: Comparison of detection rate of GMF feature and FFV feature when sampling time is 0.01 s

As shown in Fig. 4, when the sampling time is 0.01 s, we can see that in terms of detection rate, with the increase of epochs, our proposed GMF features converge faster than FFV statistical features, and the GMF feature has a higher detection rate, reaching about 94%, but FFV feature has only 85%. Thus, when the sampling time is 0.01 s, the number of data packets of unit time is less, and the multi-scale convolution model can extract more microscopic features.

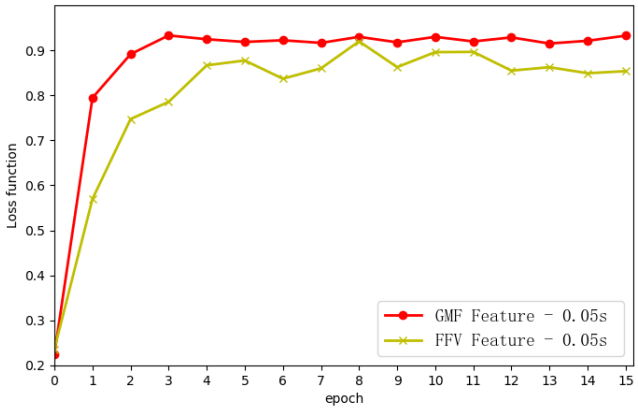

Figure 5: Comparison of detection rate of GMF feature and FFV feature when sampling time is 0.05 s

As shown in Fig. 5, when the sampling time is 0.05 s, with the increase of epochs, our proposed GMF features converge faster than FFV statistical feature, and the GMF feature has a higher detection rate, reaching about 94%.

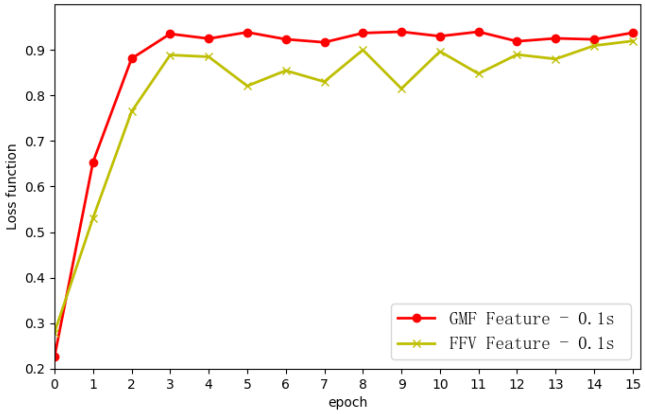

Figure 6: Comparison of detection rate of GMF feature and FFV feature when sampling time is 0.1 s

As shown in Fig. 6, when the sampling time is 0.1 s, we can see that in terms of detection rate, with the increase of epochs, our proposed GMF feature converge faster than FFV statistical feature, and the GMF feature has a higher detection rate, reaching about 93%, while we can see that the statistical characteristics of FFV oscillate obviously with the

increase of iterations in the training process. We can find that GMF feature has higher detection rate and model adaptability.

Table 1: Comparison results of different features evaluation Indicators in the change of sample time

Feature		GMF			FFV		
Sample Time (s)		0.01	0.05	0.1	0.01	0.05	0.1
Multi-Scale model	DR (%)	94.87	94.19	93.70	90.78	90.75	90.71
	FR (%)	3.78	4.53	6.23	7.32	6.59	7.32
	ER (%)	3.12	6.19	6.28	7.28	7.33	7.28
SVM	DR (%)	90.21	88.42	87.52	89.04	87.63	86.47
	FR (%)	9.93	10.25	16.86	13.33	14.28	15.26
	ER (%)	9.79	11.55	12.57	9.60	11.36	11.51
KNN	DR (%)	88.60	87.92	81.07	84.76	82.42	73.09
	FR (%)	6.90	8.89	17.24	9.63	9.13	25.00
	ER (%)	11.43	12.07	18.93	6.53	9.19	26.89
RF	DR (%)	90.33	90.55	87.92	87.65	88.79	88.93
	FR (%)	5.28	6.97	6.70	10.53	7.54	6.91
	ER (%)	5.19	6.36	6.75	10.34	7.28	8.00

As shown in Tab. 1, we can see that our proposed GMF feature has a better detection rate, lower false alarm rate and lower total error rate under multi-scale convolution model. From there we can see that Statistical-based features will result in some information missing due to statistical steps, which makes the extracted features unable to fully reflect the characteristics of network flow. The GMF feature matrix, because of binary preprocessing of the original data, arranges the binary data in the form of a high-dimensional matrix. It is more obvious to characterize the characteristics of network flow and better reflect the distribution of data packet attributes.

5.2.2 Comparison of multi-scale model and CNN model

(1) Comparison of multi-scale model and CNN model when the sampling time is 0.01 s.

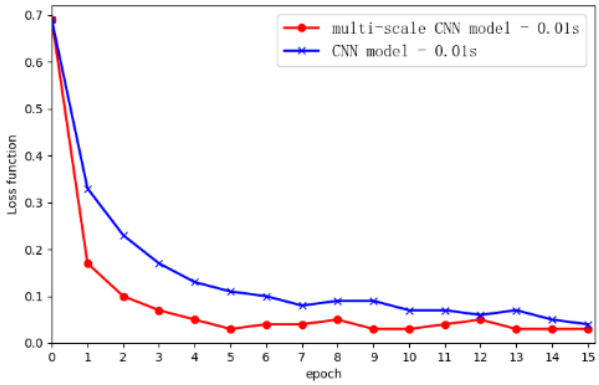

Figure 7(a): The loss function of multi-scale model and CNN model

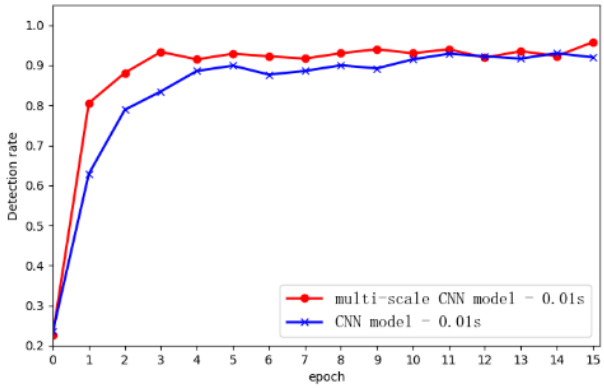

Figure 7(b): The detection rate of multi-scale model and CNN model

Figure 7: Comparison of multi-scale model and CNN model when sampling time is 0.01 s

As shown in the Fig. 7, when the sampling time is 0.01 s, the convergence speed of the proposed multi-scale optimization model is faster than that of the general model. The loss function of the multi-scale optimization model reaches convergence at the third epoch, while that of the non-optimized model reaches convergence at the sixth epoch. We can conclude that the multi-scale convolution neural network model can better adapt to the GMF features we proposed, and the training speed is faster.

(2) Comparison of multi-scale model and CNN model when the sampling time is 0.05 s.

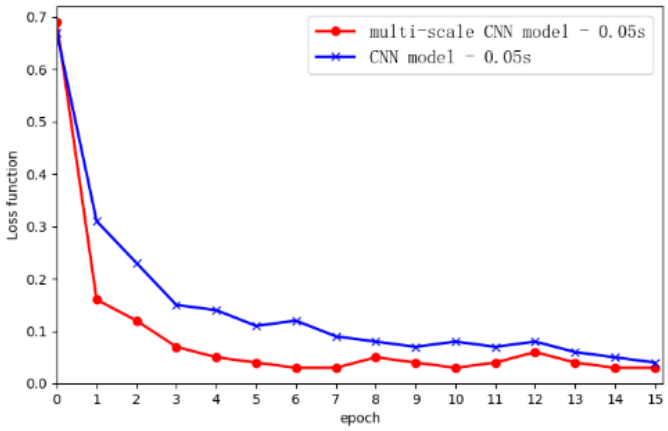

Figure 8(a): The loss function of multi-scale model and CNN model

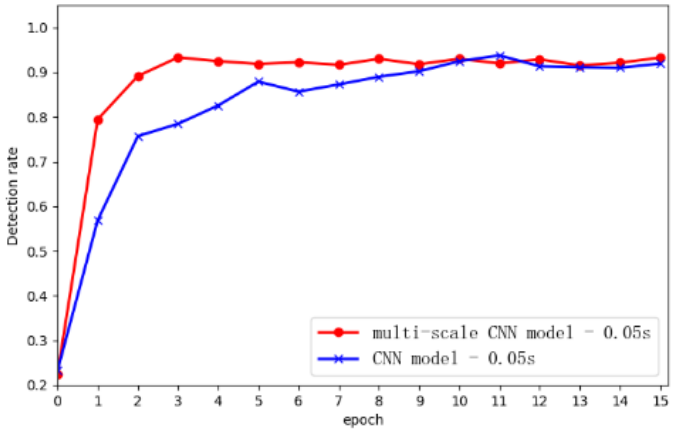

Figure 8(b): The detection rate of multi-scale model and CNN model

Figure 8: Comparison of multi-scale model and CNN model when sampling time is 0.05 s

As shown in Fig. 8, when the sampling time is 0.05 s, the convergence speed of the proposed multi-scale optimization model is faster than that of the conventional CNN model. The loss function of the multi-scale optimization model converges in the third epoch, while the loss function of the non-optimization model converges in the eighth epoch. We can conclude that the training speed of multi-scale convolution neural network model is faster and the convergence is more stable. Therefore, the multi-scale model can better adapt to the GMF feature.

(3) Comparison of multi-scale model and CNN model when the sampling time is 0.1 s.

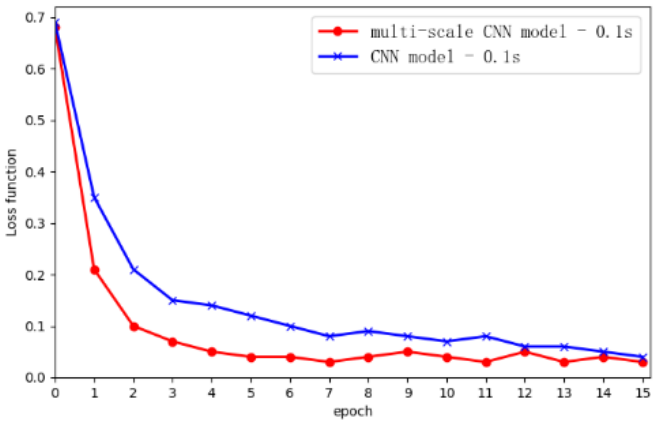

Figure 9(a): The loss function of multi-scale model and CNN model

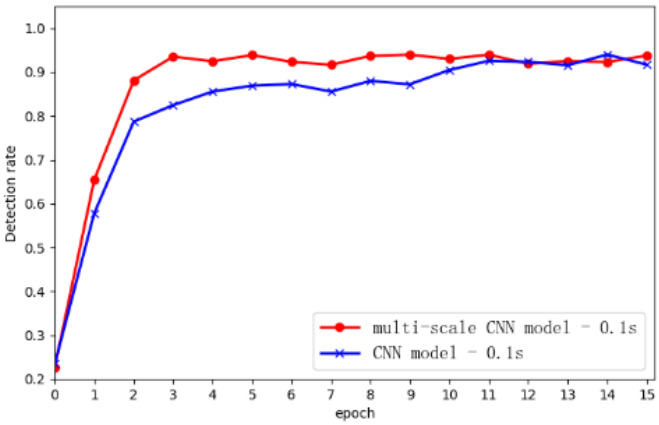

Figure 9(b): The detection rate of multi-scale model and CNN model

Figure 9: Comparison of multi-scale model and CNN model when sampling time is 0.1 s

As shown in Fig. 9, when the sampling time is 0.1 s, the convergence speed of the multi-scale optimization model is faster than that of the conventional CNN model. The loss function of the multi-scale optimization model converges in the third epoch, while the loss function of the non-optimization model converges in the seventh epoch. We can conclude that the training speed of multi-scale convolution neural network model is faster and the convergence is more stable. Therefore, the multi-scale model can better adapt to the GMF feature.

By comparing the above three sampling time models, we can see that the proposed multi-scale convolution neural network model has faster convergence speed and better model stability in the training process.

Table 2: Comparison results of multi-scale model and CNN model

Feature		Multi-scale model			CNN model		
Sample Time (s)		0.01	0.05	0.1	0.01	0.05	0.1
GMF feature	DR (%)	94.87	94.19	93.70	92.87	92.94	91.31
	FR (%)	3.78	4.53	6.23	4.78	4.76	5.66
	ER (%)	3.12	6.19	6.28	5.32	6.14	5.47

Tab. 2 shows the performance comparison of GMF features under the multi-scale CNN model and CNN model. We can see that the proposed method has a higher detection rate than other methods, reaching 94.87%. Multi-scale model has better detection performance for our proposed features. Compared with CNN methods, the proposed method has lower false alarm rate and total false alarm rate. Therefore, the CNN method has better performance in extracting the features of multi-dimensional matrix. By optimizing the parameters of the model, we build a multi-scale kernel model, which has better adaptability to the GMF matrix proposed in this paper.

6 Conclusion

Aiming at the problem of false alarm rate and missing alarm rate in DDoS attack detection methods in big data environment, we propose a DDoS attack detection method based on convolutional neural network. Based on GMF, the convolution layer at different spatial scales to imporve the segmentation accuracy, the global and local features of the network flow are extracted to resist over-fitting and improve computational efficiency. The network flow isomorphism output of the full connectivity layer is sent to the softmax classifier to take advantage of the contextual relationship of the features to improve classification accuracy. The classifier is trained by normal samples and DDoS attack samples to obtain optimal network parameters, and a DDoS attack classifier based on multi-scale convolutional neural network is constructed. Experiments show that this method has higher accuracy than similar detection methods, reduces false alarm rate and lost alarm rate, and it can effectively detect DDoS attacks under big data.

Acknowledgement: This work was supported by the Hainan Provincial Natural Science Foundation of China [2018CXTD333, 617048]; National Natural Science Foundation of China [61762033, 61702539]; Hainan University Doctor Start Fund Project [kyqd1328]; Hainan University Youth Fund Project [qnjj1444].

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Ahanger, T. A. (2017): An effective approach of detecting DDoS using artificial neural networks. *International Conference on Wireless Communications, Signal Processing and Networking*, pp. 707-711.

Behal, S.; Kumar, K. (2017): Characterization and comparison of DDoS attack tools and traffic generators: a review. *International Journal of Network Security*, vol. 19, no. 3, pp. 383-393.

Behal, S.; Kumar, K. (2017): Detection of DDoS attacks and flash events using novel information theory metrics. *Computer Networks*, vol. 116, pp. 96-110.

Cheng, J.; Liu, B.; Tang, X. (2018): An automatic traffic-congestion detection method for bad weather based on traffic video. *International Journal of High Performance Computing and Networking*, vol. 11, no. 3, pp. 251-259.

Cheng, R.; Xu, R.; Tang, X.; Sheng, V. S.; Cai, C. (2018): An abnormal network flow feature sequence prediction approach for DDoS attacks detection in big data environment. *Computers, Materials & Continua*, vol. 55, no. 1, pp. 95-119.

Cheng, J.; Yin, J; Liu, Y.; Cai, Z.; Li, M. (2009): DDoS attack detection algorithm using IP address features. *International Workshop on Frontiers in Algorithmics*, vol. 5598, pp. 207-215.

Cheng, J.; Zhang, C.; Tang, X.; Sheng, V. S.; Dong, Z. et al. (2018): Adaptive DDoS attack detection method based on multiple-kernel learning. *Security and Communication Networks*, vol. 2018.

Gao, C. Z.; Cheng, Q.; He, P.; Susilo, W.; Li, J. (2018): Privacy-preserving naive bayes classifiers secure against the substitution-then-comparison attack. *Information Sciences*, vol. 444, pp. 72-88.

Ham, F. M.; Kostanic, I. (2000): *Principles of Neurocomputing for Science and Engineering*. McGraw-Hill Higher Education.

Idhammad, M.; Afdel, K.; Belouch, M. (2018): Detection system of http DDoS attacks in a cloud environment based on information theoretic entropy and random forest. *Security and Communication Networks*, vol. 2018.

Mamolar, A. S.; Pervez, Z.; Calero, J. M. A.; Khattak, A. M. (2018): Towards the transversal detection of DDoS network attacks in 5G multi-tenant overlay networks. *Computers & Security*, vol. 79, pp. 132-147.

Mirkovic, J.; Reiher, P. (2004): A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39-53.

Pandey, V. C.; Peddoju, S. K.; Deshpande, P. S. (2018): A statistical and distributed packet filter against DDoS attacks in cloud environment. *Sādhanā*, vol. 43, no. 3, pp. 32.

Petkovic, M.; Basicevic, I.; Kukolj, D.; Popovic, M. (2018): Evaluation of Takagi-Sugeno-Kang fuzzy method in entropy-based detection of DDoS attacks. *Computer Science and Information Systems*, vol. 15, no. 1, pp. 139-162.

Pras, A.; Santanna, J. J.; Steinberger, J.; Sperotto, A. (2016): DDoS 3.0-how terrorists bring down the internet. *International GI/ITG Conference on Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, pp. 1-4.

Saied, A.; Overill, R. E.; Radzik, T. (2016): Detection of known and unknown DDoS attacks using artificial neural networks. *Neurocomputing*, vol. 172, pp. 385-393.

Suzuki, K.; Murayama, N. (1985): Conversion of multilevel digital data to binary data. US Patent 4,562,486.

Wang, Y.; Ma, J.; Zhang, L.; Ji, W.; Lu, D. et al. (2016): Dynamic game model of botnet DDoS attack and defense. *Security and Communication Networks*, vol. 9, no. 16, pp. 3127-3140.

Wang, C.; Zheng, J.; Li, X. (2017): Research on DDoS attacks detection based on RDF-SVM. *10th International Conference on Intelligent Computation Technology and Automation*, pp. 161-165.

Yadav, V. K.; Trivedi, M. C.; Mehtre, B. (2016): DDA: an approach to handle DDoS (ping flood) attack. *Proceedings of International Conference on ICT for Sustainable Development*, pp. 11-23.

Ye, J.; Cheng, X.; Zhu, J.; Feng, L.; Song, L. (2018): A DDoS attack detection method based on SVM in software defined network. *Security and Communication Networks*, vol. 2018.

Yuan, X.; Li, C.; Li, X. (2017): DeepDefense: identifying DDoS attack via deep learning. *IEEE International Conference on Smart Computing*, pp. 1-8.

Yu, J.; Hu, M.; Wang, P. (2018): Evaluation and reliability analysis of network security risk factors based on ds evidence theory. *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 2, pp. 861-869.