Hindawi
Security and Communication Networks
Volume 2022, Article ID 7551107, 18 pages
https://doi.org/10.1155/2022/7551107

Research Article

All-Packets-Based Multi-Rate DDoS Attack Detection Method in ISP Layer

Xinqian Liu ,[1] Jiadong Ren ,[2] Haitao He,[2] Bing Zhang,[2] Qian Wang,[2] and Zhangqi Zheng[2]

[1]*School of Computer Science and Technology, Shandong University of Technology, Zibo 255000, Shandong, China*
[2]*School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, Hebei, China*

Correspondence should be addressed to Jiadong Ren; jdren@ysu.edu.cn

Received 27 January 2022; Accepted 4 April 2022; Published 6 May 2022

Academic Editor: Zhen Wang

Distributed denial of service (DDoS) is a fundamental security problem in the ISP layer of the internet of things. However, most existing DDoS detection methods are based on NetFlow data, which cannot handle the huge detection delay of flow generation and massive network traffic. Besides, it is extremely hard to obtain the real DDoS attack traffic to construct a traditional supervised binary classification model. To solve these problems, this paper proposes a novel all-packets-based DDoS attack detection method (APDD). Firstly, a new probabilistic storage model square sketch is designed, which has structural characteristics of parallelization, accumulation, and recompression. The model and its characteristics are conducive to fast and efficient traffic storage and compression. All network packets are mapped into square sketch, and the compressed square sketch is obtained. Secondly, in order to overcome the problem of poor real DDoS attack samples, only according to the recompressed square sketch of the normal network, a one-class classifier is constructed by generative adversarial networks to form a DDoS attack detection model. The likelihood score of a recompressed square sketch is obtained to judge this square sketch whether or not it belongs to a normal network state. Finally, two real network traffic data sets of the high-throughput network are utilized to evaluate the proposed method. Compared with several existing methods, the experimental results show that the APDD method has good time efficiency and detection performance.

1. Introduction

The internet of things (IoT) is a huge association between computing devices, mechanical devices, digital devices, and human entities. These entities cooperate with each other by manufacturing, collecting, and processing data to provide users with an intelligent adaptive environment, such as smart home environment, intelligent transportation system, intelligent security network, and so on. There will be more than 21 billion internet of things devices by 2025. Unfortunately, the security of internet of things devices cannot keep up with the rapid development of internet of things applications. Now more and more vulnerabilities are detected regularly, which result in security threats and privacy problems. For example, such compromised devices can be used to perform distributed denial of service (DDoS) attacks [1, 2]. The Mirai botnet triggered the largest number of DDoS attacks since 2016. This special botnet infects many internet of things devices (mainly older routers and IP cameras). And the data transmission rate exceeds 600 Gbps. Mirai botnet destroyed many popular websites such as Etsy, Netflix, Shopify, SoundCloud, and Twitter by injecting traffic into DNS providers. In March 2018, GitHub, a well-known code-hosting website, suffered from the most serious DDoS attack ever with 1.35 Tbps peak traffic. This attack suddenly interrupted network services to cause huge economic losses. DDoS attacks are regarded as the most critical threat to the operation of individual enterprises and organizations and the stability of the whole Internet. From the development of DDoS attacks, the TB level of DDoS attacks will gradually become normal, and the serious consequences are immeasurable [3]. There are many kinds of occurrences and expressions of DDoS attacks [4]. From the perspective of occurrence way, DDoS attacks can be generated by botnet, proxy, or spoofing IP [5]. From the perspective of expression way, DDoS attacks are divided into high- and low-rate DDoS attacks [6]. Focusing on this attack, it is difficult to effectively and quickly detect and defend against DDoS attacks [7].

The convenience provided by the internet of things has led to the extensive deployment of various sensors, such as thermostats, security cameras, and smart lights. The number of the internet of things devices is growing exponentially, and the transmitted network traffic is also growing exponentially [8]. Therefore, the high-throughput network environment brings great challenges to DDoS attack detection. In the 100 Gbps network, the number of network flows simultaneously reaches 50 million [9]. Therefore, it is impossible to collect, store, and process the whole network traffic. To solve this problem, some commercial tools, such as NetFlow and CFlow, are designed to reduce data storage space by integrating packets into flows. These tools are currently deployed on some routers and switches [10]. However, the integration mechanism increases the intermediate steps of network traffic processing and results in the delay of network anomaly detections [11]. Besides, there exists an effective alternative strategy to only collect and process the packet header [12]. However, the volume of packet headers is still very huge. For example, the packet header of ethernet at least is 224 bits. When there are 1 million packets per second, the daily storage of packet header information will exceed 1 T. Though based on packet headers or NetFlow data, many traditional detection methods bring great challenges to the performance of memory and CPU by frequently accessing and calculating a large amount of data. It is also impossible to achieve online detection [13]. The attack detections have to be carried out in an offline way after a few minutes or even hours. This kind of detection delay is fatal for system security when DDoS attacks occur [14]. In addition, due to a high sampling ratio (e.g., the sampling ratio is 1:1,000) of NetFlow or CFlow tools, low-rate attacks are easily evaded [15]. Hence, it is more difficult to detect low-rate DDoS attacks. And the high sampling ratio results in a high false-negative rate.

Focusing on the above problems, the probabilistic data structure, as effective data compression and storage technology, has been applied to handle large-scale network traffic. This technology has been widely applied in the field of network anomaly detection. The sketch is the most popular probabilistic data structure, which effectively compresses data through hash functions to reduce data storage and analysis consumption [16, 17]. For example, the sketch has the ability to compress thousands of megabits of data into tens of megabits. Wang et al. [18] utilized sketch and an improved Hellinger distance to detect application-layer DDoS attacks. Jing et al. [19] proposed a reversible sketch based on the Chinese remainder theorem to map network traffic. And the mutation of one-to-one mapping between request and response packets was monitored to identify amplification attacks. Therefore, the probabilistic data structure is an effective way to deal with massive network traffic. It is possible to use the probabilistic data structure to manage all network packets and avoid the detection error caused by the high ratio sampling.

Another important challenge is the obtention of DDoS attack traffic in the real network environment [20]. Normal network traffic is common and easy to obtain. Hence, it is very difficult to build a binary supervised model to detect DDoS attacks. As mentioned above, one kind of data is easy to obtain, while negative class data is difficult to obtain and label. This situation promotes the development of one-class classification (OCC) model that can well distinguish the target class and nontarget class [21]. At present, OCC methods are widely applied to outlier detections, image denoising, and image anomaly detections [22]. To our knowledge, OCC methods are rarely applied to DDoS attack detections. Therefore, in the absence of DDoS attack traffic, it is necessary to introduce the OCC model to build an attack detection model.

Facing these challenges, this paper proposes a lightweight DDoS attack detection method based on all packets, which realizes a fast and efficient DDoS attack detection in a high-throughput network. Different from the previous flow-based solutions, the whole packets are processed without sampling. For all packets, instead of the traditional sketch structure, a new sketch structure square sketch is proposed. The update rule and two independent hash functions with keys of the IP pair are defined. Furthermore, the structure characteristics (parallelization, accumulation, and recompression) of the square sketch are fully revealed, which is helpful to achieve faster compression and smaller storage. Then, an adversarial one-class learning model is designed, in which the convolution layer is the learning unit. A compressed square sketch is intuitively a two-dimensional matrix and similar to the image form. Hence, it has good adaptability to be the input of the adversarial one-class learning model. The designed adversarial one-class learning model can realize effective DDoS attack detection. The main contributions of this paper are as follows:

(1) This paper proposes an all-packets-based DDoS attack detection model in the high-throughput network to detect high- and low-rate DDoS attacks. The method not only realizes the real-time detection requirements but also meets the detection accuracy requirements of different rates of DDoS attacks.

(2) An all-packets data mapping method based on a square sketch is built, which avoids the intermediate steps of network traffic handling and the negative impact of sampling. By analyzing the structural characteristics of a square sketch, the faster data compression and smaller data representation are obtained. The processing is helpful for quick attack detections.

(3) The viewpoint of one-class learning model is introduced. And an adversarial one-class classification model is constructed by an automatic encoder and a discriminator. The model is perfectly combined with the square sketch to realize effective DDoS attack detections.

The organization of this paper is as follows. Second 2 introduces the related DDoS attack detection methods. Section 3 describes the overall description and detailed process of the proposed DDoS attack detection method. Section 4 shows the experimental setup and performance evaluation. Section 5 is the summary of this work.

2. Related Work

This section will describe the existing DDoS attack detection methods in the ISP layer. However, the related researches are less. Gupta et al. [23] proposed a DDoS attack countermeasure scheme at the ISP layer to monitor the propagation of traffic mutations. Using two statistical metrics of volume and flow as parameters, Six Sigma and variable tolerance factor methods were used to accurately and dynamically identify the thresholds of various statistical metrics. The inaccurate threshold results in a large number of false-positive and false-negative rates. The NS-2 network simulator on the Linux platform was utilized as the simulation testbed to validate the effectiveness of this method. Different attack scenarios were achieved by changing the zombie number and attack strength. Compared with volume-based approaches, this proposed scheme has a good performance.

Hinze et al. [24] utilized a passive measurement method to analyze malicious traffic on the national ISP and large regional internet exchange points. According to the MAWI data set on the ISP layer, the method identified DDoS attacks by analyzing the statistical rules of IP address, transport layer type, source port, TCP flag, and so on. The experiment results displayed that the false alarm rate of this method is about 20%~70%. Therefore, this method needs to be improved to find DDoS attacks well. However, this paper illustrated that the MAWI data set is a basis data set to analyze DDoS attacks on the ISP layer.

Liu et al. [25] developed a multi-layer defense architecture to defend against various DDoS attacks. In particular, the flood throttling layer stops amplification-based DDoS attacks. The user-specific layer allowed DDoS victims to enforce self-desired traffic control policies. Based on Linux implementation, the method was capable to deal with large-scale attacks involving millions of attack flows. Furthermore, the physical testbed experiments and large-scale simulations proved that the method is effective to mitigate various DDoS attacks.

Gong et al. [26] proposed a DDoS attack detection model based on a quantum genetic optimized BP neural network (DQGA-BP). Firstly, aiming at the problem of insufficient search ability of quantum genetic algorithm (QGA), an improved QGA method was proposed, which dynamically changes the rotation angle of a quantum revolving gate. Next, the improved QGA was combined with BP neural network to detect DDoS attacks on the KDD cup 1999 data set. The experimental results showed that the improved QGA has a faster convergence speed and a stronger optimization ability. The average detection rate of DQGA-BP is 0.51491% higher than that of the original quantum genetic optimized BP neural network.

Ko et al. [27, 28] pointed out that the existing detection mechanisms are not instant. In ISP networking, the acquisition of NetFlow data is hierarchical and will take a lot of time. For implementing a fast mitigation mechanism after a few minutes of DDoS attacks, a stacked self-organizing map model was proposed to combat new DDoS attacks based on NetFlow data. This model utilized the Apache Spark framework to achieve a fast and simple calculation. Meanwhile, a dynamic network traffic management (DNTM) system was constructed, including an attack detector, an IP prioritizer, a traffic manager, and a NetFlow classifier. The traffic manager utilized the existing ISP mechanisms (including entry and exit filtering, rate limiting, black hole, and normal routing) to take different mitigation measures.

3. The Proposed Method APDD

The overall framework of the multi-rate DDoS attack detection method based on all packets in the ISP layer is shown in Figure 1. This method mainly includes three stages: data preprocessing, all packets mapping model based on the square sketch, and DDoS attack detection model based on adversarial one-class classifier. In the data preprocessing stage, the Wireshark tool is used to analyze network packets. The basic packet characteristics including timestamp, source IP address, and destination IP address are obtained. According to the timestamp information, the network packets of a unit period are input into the all-packets mapping model based on the square sketch. The square sketch of one period is obtained by hash mapping through n threads. Then, according to the compression unit c, the square sketch is compressed to get a compressed square sketch (CSS). Next, the historical normal CSSs are the input of the adversarial one-class classification (AOCC). Via these CSSs, the AOCC model is trained to obtain the optimal DDoS attack detection model. Finally, the CSS of the current time period is input into the trained AOCC model, and the current detection result is obtained. It is worth noting that the AOCC model is not fixed, but it is trained and updated periodically according to historical CSSs data, which is more suitable for the fluctuation of network traffic.

3.1. All-Packets Data Mapping Model via Square Sketch.

In this section, the square sketch structure and the process of all packets processing are described in detail. Firstly, the new structure and update rule of the proposed sketch are described. Then the structural characteristics of the square sketch are analyzed, and finally, the mapping process of all packets based on the square sketch is shown.

Definition 1. Square sketch. The square sketch is a $K * K$ matrix $\{SS\}_{K*K}$. Different from the traditional sketch, the keys of the square sketch are composed of a key pair (k_1, k_2). The keys k_1 and k_2 correspond to two independent hash functions f_1 and f_2, respectively. The $f_1(k_1) \in (1, 2, \ldots, K)$ and $f_2(k_2) \in (1, 2, \ldots, K)$

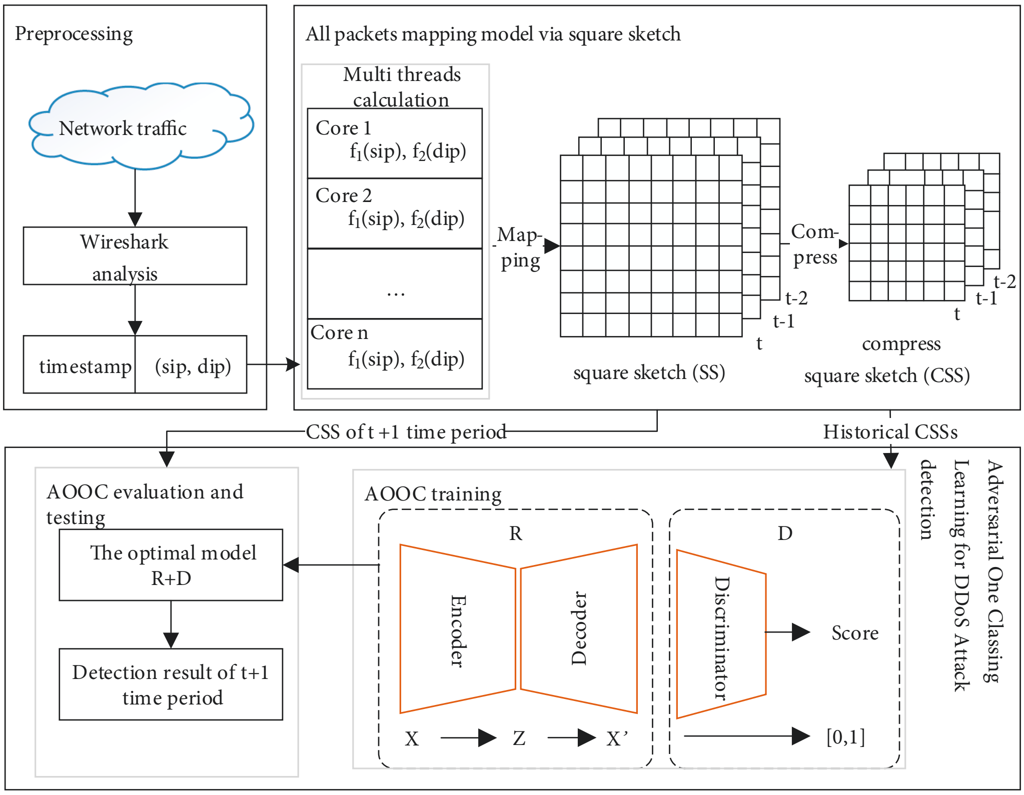


Figure 1: Overall framework of all-packets-based multi-rate DDoS attack detection method in ISP layer.

determine the corresponding position of the key values k_1 and k_2 in the square sketch.

3.1.1. Square Sketch Update. The update process of the square sketch is $i:=f_1(k_1)$, $j:=f_2(k_2)$, $SS(i,j):=SS(i,j)+v$, where v is the corresponding update value. The square sketch structure and update process are shown in Figure 2.

3.1.2. Conflict Rate of Square Sketch. When two or more packets are mapped to the same bucket, the conflict occurs. Although the sketch structure itself allows certain errors, when the error is large, the accuracy of the square sketch is seriously affected. Suppose the data amount is H, and each data is randomly mapped into a bucket by hash functions. Given an arbitrary bucket and arbitrary data, the probability that the data is mapped to the bucket is $1/K^2$. Therefore, for any bucket, H data are mapped into the bucket Z times, and the Z value obeys the binomial distribution $B(H, 1/K^2)$. When H is large and $1/K^2$ is small, and Z obeys the Poisson distribution $\pi(H/K^2)$, then

$$\Pr\{Z = i\} = e^{-H/K^2}\frac{\left(H/K^2\right)^i}{i!}. \tag{1}$$

For a bucket g, if $Z \geq 2$, there is a conflict in the bucket g. Hence, the conflict rate is

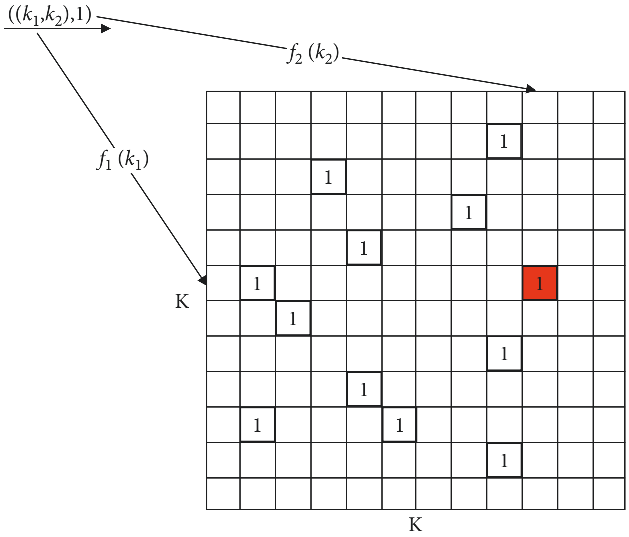


Figure 2: Square sketch structure and update process.

$$P_{cr} = 1 - P_r\{Z = 0\} - P_r\{Z = 1\} = 1 - \left(\frac{H}{K^2} + 1\right)e^{-H/K^2}. \tag{2}$$

When H/K^2 is 0.1 or 0.01, the conflict rate is 0.0046 and 0.00005, respectively. In order to illustrate the setting of K value in a real network environment, three different network environments are set, in which the number of active hosts is

Table 1: Corresponding relationship between H and K at $P = 0.0046$ and 0.00005.

H	K	
	$P = 0.0046$	$P = 0.00005$
1,000	100	317
8,000	283	894
40,000	632	2,000

500, 4,000, and 20,000. In the most random communication, the number of different IP addresses is twice the number of active hosts; then $H = 1{,}000$, 8,000, and 40,000. When the conflict rate is 0.0046 and 0.00005, the results of K value are shown in Table 1.

Square sketch has the following characteristics: parallelization, accumulation, and recompression. This structure is conducive to achieving a faster and lighter network traffic monitoring and detection in a higher speed network.

3.1.3. Parallelization. Different from the traditional network traffic processing and measurement methods (e.g., information entropy) that cannot achieve parallel processing in a single time period. The square sketch can compress network packets in a single time period by parallel processing. Figure 3 shows the execution ways of the single thread and multiple threads of the packets mapping process.

3.1.4. Accumulation. The accumulation characteristic of the square sketch can achieve to easily adjust detection periods. This characteristic has a good application in the large-scale network, which can deal with problems of excessive memory consumption and even memory overflow caused by massive network traffic over a long time. The square sketch data of different time periods can be obtained by accumulating. The accumulation formula is as follows:

$$SS_n = \sum_{i=1}^{n} SS_i, \tag{3}$$

where n is the number of time periods. The accumulation process is shown in Figure 4.

3.1.5. Recompression. In a high-speed network environment, in order to make the mapping conflict of network traffic as small as possible, the square sketch should keep a certain large size. The larger the size of the sketch structure is, the larger the size of internal storage is, and the larger the memory and time consuming of the network state discrimination is. This is very unfavorable for monitoring high-speed networks. Therefore, it is necessary to compress the square sketch again to reduce the memory consumption and improve the efficiency of network monitoring. Compression square sketch mainly includes two main steps: grouping and merging.

In order to realize the grouping operation, a compression rate c is defined. For the sake of brevity, c is best divisible by K. The grouping process is as follows. (1) A $K * K$ square sketch A is equally divided into $w = K/c$ parts in rows and columns and get w^2 groups. The size of each group is c^2. (2) A new square sketch B is constructed with the size of $w * w$. (3) The data of each group $A_{ij}^{mn}, m, n \in [1, 2, \cdots, w], i, j \in [1, 2, \cdots, c]$ are merged into the corresponding bucket of B and obtain $B_{mn} = OP_{i,j=1}^{c}\{A_{ij}^{mn}\}$. In the process of merging operation, two merging methods are proposed: SUM operation and MAX operation. SUM operation refers to the add operation of data on each group, $B_{mn} = \mathrm{SUM}_{i,j=1}^{c}\{A_{ij}^{mn}\}$. MAX operation refers to selecting the maximum value of each group data as the combined value, $B_{mn} = \mathrm{MAX}_{i,j=1}^{c}\{A_{ij}^{mn}\}$. The SUM and MAX operations are shown in Figure 5.

In this paper, SUM operation is chosen as the merge operation because the MAX operation is not always valid. When the detection period $k = 1$, MAX merging operation is performed on SS. Due to $SS_{ij} \in \{0, 1\}, i, j = 1, 2, \ldots, K$, $B_{mn} = \mathrm{MAX}_{i,j=1}^{c}\{A_{ij}^{mn}\} = \{0, 1\}$. The value of CSS in all time windows is 0 or 1, which is not discriminative. When $k > 1$, the detection period k is usually not very large. Assume that $1 < k < 10$, $B_{mn} = \mathrm{MAX}_{i,j=1}^{c}\{A_{ij}^{mn}\} = \{0, 1, \cdots, k\}$. The maximum value of CSS in the attack state may be k. The CSS in the normal situation may also produce the maximum value k. Therefore, CSS with MAX operation does not have the ability to identify attacks and is invalid. When DDoS attacks occur, compared with the traditional count-min sketch, which concentrates the attack data in a certain number of point data, the square sketch concentrates the attack data in a certain column (one attack target) or several columns (multiple attack targets). Hence, the square sketch can better show the difference between the normal sketch and the attack sketch and is more conducive to the subsequent attack detection processing.

The detailed process of all packets mapping models based on the square sketch is described. Network traffic can be regarded as a continuous stream $(w_n, v_n), n = 1, 2, \ldots$. The w_n refers to the key value, which can be the IPv4 address, $w_n \in \Omega = \{0, 1\}^{32}$. The v_n refers to the corresponding update value, which can be the number of bytes or packets. In this paper, w_n refers to the key pair (sip_n, dip_n) composing of source IP address and destination IP address, and v is equal to 1. In order to monitor all packets, the update rule of the square sketch is slightly different from the initial rule, shown in formula (4). Compared with the 1-universal hash function, the applied 4-universal hash function can avoid the mapping conflict to a greater extent, as shown in formula (5).

$$SS(f_1(sip), f_2(di\ p)) = \begin{cases} 1 & \text{if } SS(f_1(sip), f_2(di\ p)) == 1 \\ 1 & \text{if } SS(f_1(sip), f_2(di\ p)) == 0 \end{cases}, \tag{4}$$

$$f_i(\text{key}) = \sum_{j=0}^{3} \left(a_{ij}\text{key} + b_{ij}\right) \text{mod}\ p\ mod\ K, \quad i = 1, 2, \tag{5}$$

where p is a prime and greater than the largest key value. For a faster mapping, p is set to $2^{31}-1$, which is a Mersenne prime. $a_{ij}, b_{ij} \in [1, p-1]$ are two random values. K is the width of the square sketch.

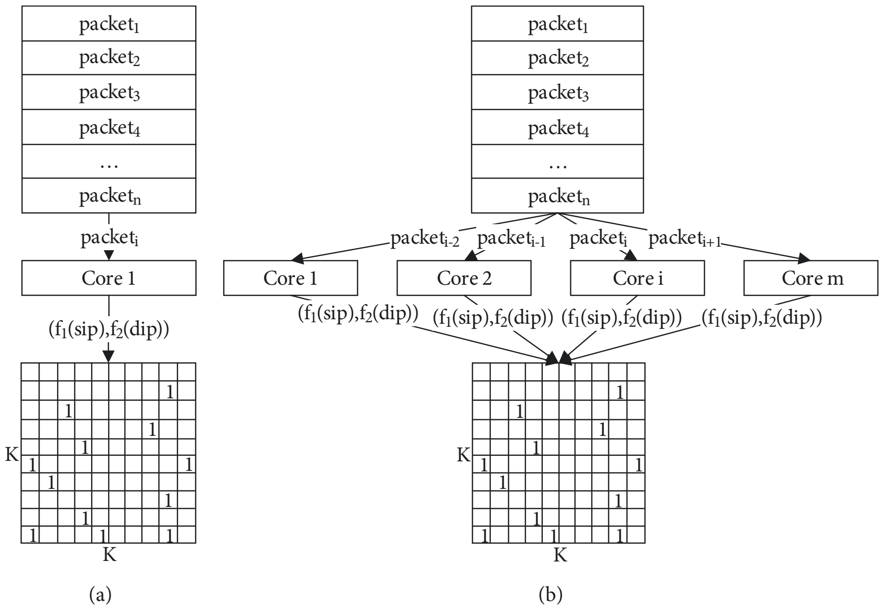


Figure 3: Single- and multi-threads execution of packets mapping process in the square sketch: (a) single-thread execution and (b) multi-threads execution.

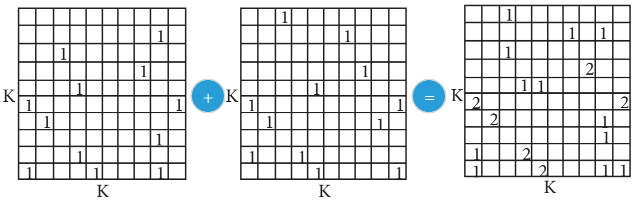


Figure 4: Accumulating results of the square sketch in $n = 2$.

The size of the square sketch is generally large in order to reduce the conflict of packet mapping, which is unfavorable for data storage and subsequent attack detections. Therefore, after packets in a unit period are mapped to a square sketch, the square sketch data is compressed to a CSS according to the grouping and merging operations. In order to realize the effective detection of CSSs, standardization is an essential step to make the compressed square sketch data in the same standard each time. The CSS standardization formula is as follows:

$$NCSS = \frac{CSS}{n * c^2}, \tag{6}$$

where n is the number of time windows, c is the compression unit, and c^2 is the size of the compression group.

For continuous data packets, all packets are mapped to a square sketch by multiple threads. Then the mapped square sketch is compressed further. The detailed process of the all-packets mapping method based on a square sketch is shown in Algorithm 1.

In Algorithm 1, according to multi-threads, all packets P in time period t are mapped by the 4-universal hash function. SS is updated according to the obtained mapping values, as shown in lines 02–06. Lines 02 and 03 build a thread pool including n threads. The subfunction *ipsketch* is executed for all packets P in a multi-thread manner. The *ipsketch* represents the mapping process of each packet. The mapping results *Bucket_List* is obtained by executing the *ipsketch* function through n threads as shown in line 03. The *Bucket_List* circularly updates SS as shown in lines 04–06. Lines 07 and 08 compress and standardize SS to obtain $NCSS$.

3.2. Adversarial One-Class Classification for DDoS Attack Detections. In this section, the square sketch is a matrix, which is different from the traditional feature vector. In particular, the sketch is considered as the image of one channel as Figure 6. Figures 6(a) and 6(b) are the visualizations of the square sketch in the normal traffic and DDoS attack traffic, respectively. It can be seen that under the normal network environment, the color distribution of pixels in the image is relatively uniform. But, when DDoS attacks appear, a column of pixels with darker color occurs. This situation indicates that a destination receives massive attack packets from multiple sources. Hence, the deep learning model is determined as the target class classification model to achieve better detection results. An adversarial one-class learning model is developed to automatically extract discriminative features and distinguish target class from nontarget class [29]. The model consists of two parts: reconstructor (R) and discriminator (D). The reconstructor model learns the important features of the target class and reconstructs the input data. The discriminator learns the features of the target class by classifying the original input data and reconstructing data, so as to achieve the purpose of discriminating against the target class. Via the end-to-end training process, the R network can reconstruct the data from the target class and distort the nontarget class. D network can output whether the reconstructed data belongs to the target class or not.

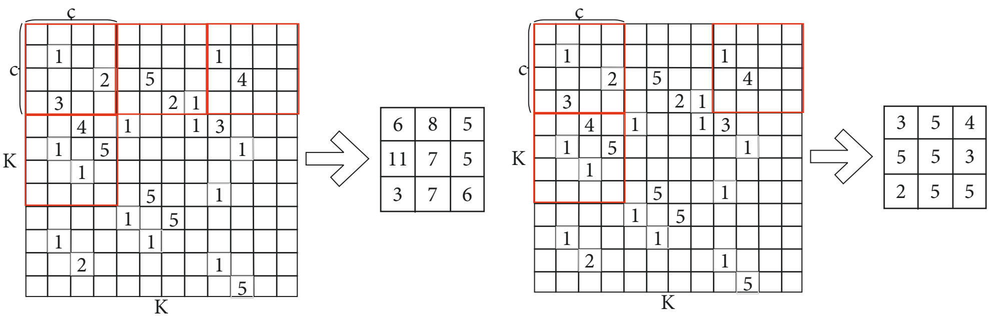
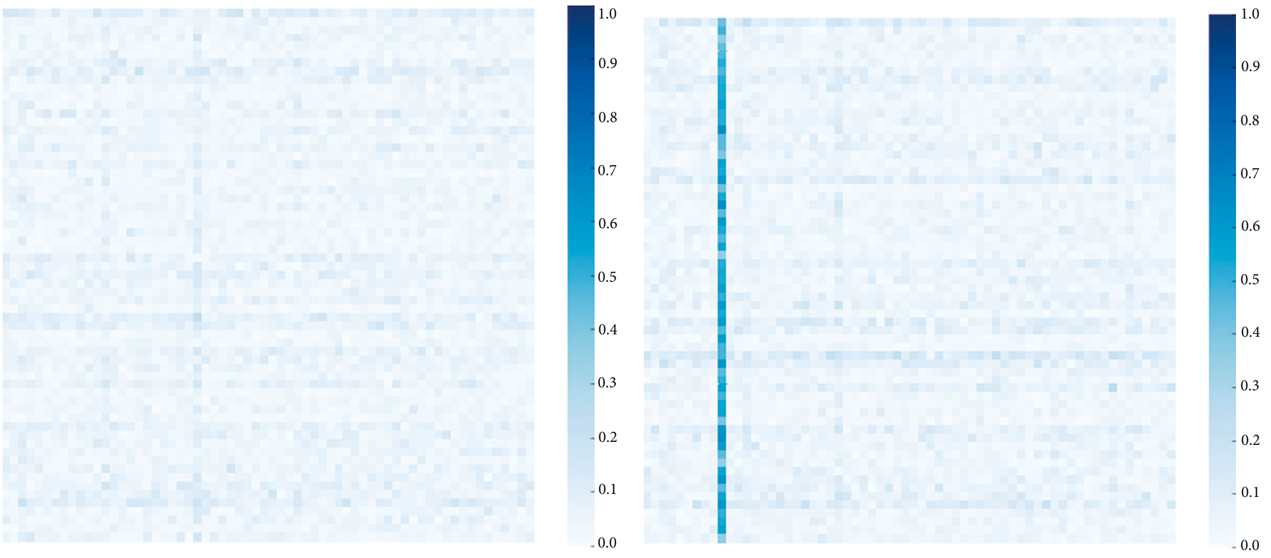


Figure 5: Merge operation: (a) SUM operation and (b) MAX operation.

```
Input: network packets of t-th period $P$, $f_1$, $f_2$, $m$, $c$
Output: NCSS of t-th period
SS, CSS = []
pool = multiprocessing.Pool(processes = m)
Bucket_List = pool.map_async(ipsketch, P).get()
For item in Bucket_List:
    SS[item[0], item[1]] = 1
End for
compress SS to obtain CSS
standardizing CSS to obtain NCSS with formula (6)
Produce ipsketch(p, $f_1$, $f_2$):
    sip = int(p.sip), dip = int(p.dip)
    i = $f_1$(sip)
    j = $f_2$(dip)
return i, j
```

Algorithm 1: All-packets data mapping model via square sketch.



(a)

(b)

Figure 6: Visualizations of square sketch: (a) normal traffic and (b) DDoS attack traffic.

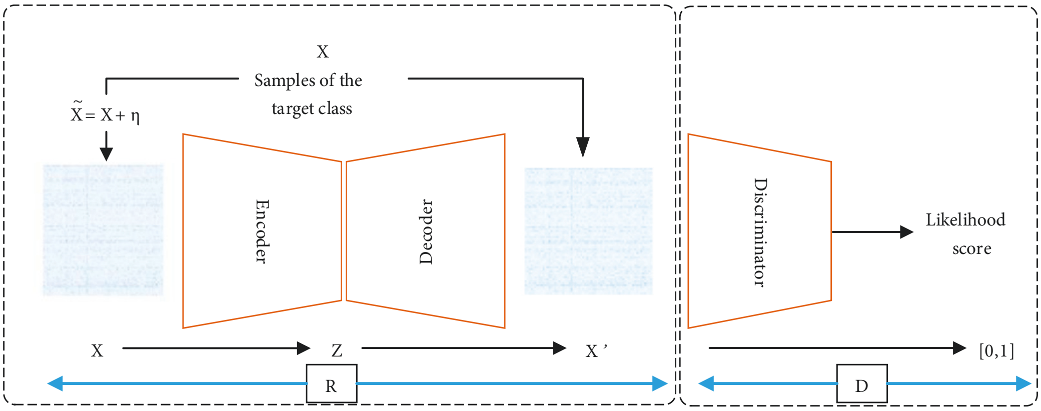


Figure 7: Framework of adversarial one-class classification model.

The model framework is shown in Figure 7. R network reconstructs input X to generate X' to deceive network D, and D learns original data X to maximumly discriminate between X and X'. In order to make the model more robust, the Gaussian noise η is added to the original data and fed back to the R network.

3.2.1. Network Architecture. In the R network, an encoding-decoding convolutional neural network is trained to realize the feature learning and reconstruction of the input data. Obviously, for the sample data similar to the target class, the R network can reconstruct very well. However, for outliers or abnormal data, the reconstructed data are very poor. Hence, it is easy for the discriminator to distinguish the target class from the nontarget class. Figure 8(a) shows the structure of the R network. The R network consists of several convolution layers (encoding) and deconvolution layers (decoding). A batch standardization layer is added after each convolution layer to maintain the model stability [30]. D network is composed of multiple convolution layers and a fully connected layer. The output of the D network is a score between 0 and 1, which is equivalent to the likelihood of the target data distribution, expressed as p_t. D network architecture is shown in Figure 8(b).

3.2.2. Model Training. The training process of the $R+D$ neural network model is divided into two steps: joint training and only automatic encoder training. The purpose of joint training is to train the discriminator to obtain the ability to identify true and false data. The only automatic encoder training is to reconstruct the target class better, distort the nontarget class, and increase the distance between the target class and the nontarget class to a certain extent. Like the traditional generative adversarial network, R and D networks are carried out in an adversarial procedure. On the contrary, instead of mapping the potential space Z to a data sample with distribution pt, R maps

$$\overline{X} = (X \sim p_t) + \left(\eta \sim N\left(0, \sigma^2 I\right)\right) \longrightarrow X' \sim p_t, \qquad (7)$$

where η is the added noise samples from the normal distribution with the standard deviation σ, $N(0, \sigma^2 I)$. For simplicity, the noise model is expressed as N_σ. In the training process, p_t is the assumed distribution of the target class. D is aware of p_t, as it is exposed to samples from the target class. Therefore, D clearly distinguishes whether $\mathscr{R}(\tilde{X})$ follows p_t. Therefore, the optimization objectives of joint training are as follows:

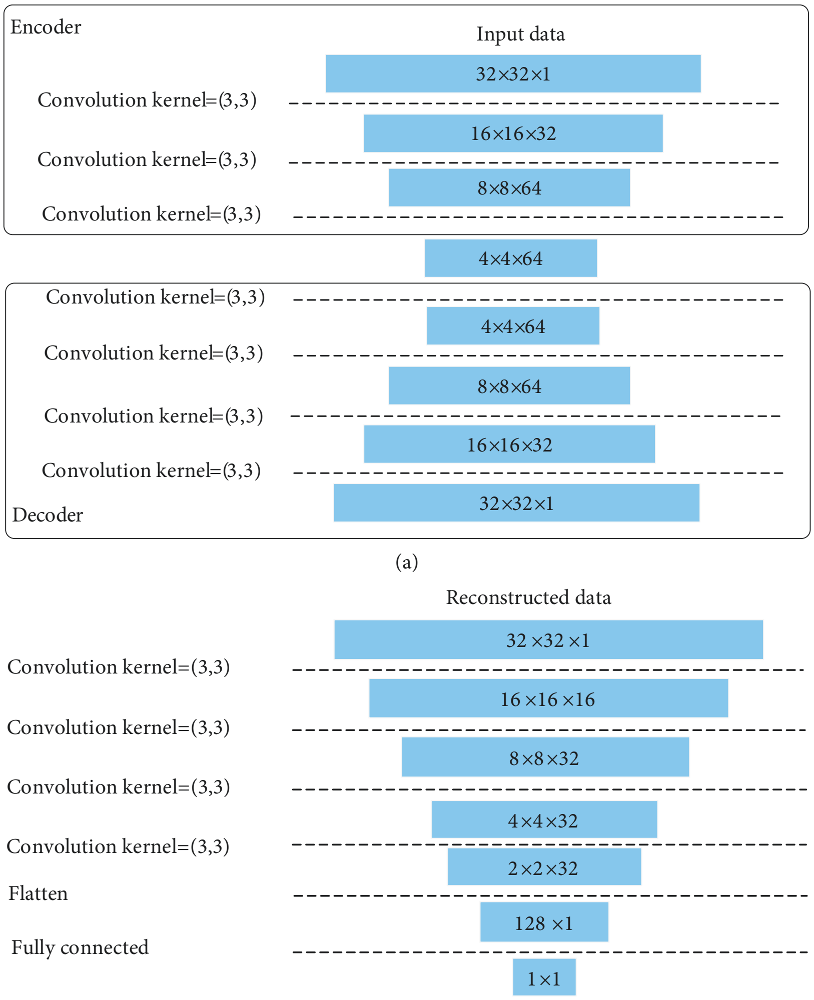
$$\min_R \max_D \left(\mathrm{E}_{X \sim p_t} [\log (D(X))] + t\mathrm{E}_{\overline{X} \sim p_t + N_\sigma} n[\log (1 - D(R(\overline{X})))] \right). \qquad (8)$$

To train the model, we calculate the loss L_{R+D} as the loss function of joint network $R+D$. In addition, the loss function of the automatic encoder is $L_R = \|X - X\|^2$. Therefore, the $R+D$ network model is optimized to minimize the loss function $L = L_{R+D} + \lambda L_R$, where $\lambda > 0$ is a tradeoff hyperparameter to adjust the relative importance of the two terms. The model training is stopped when R successfully maps the noise data to the target class distribution. Therefore, when R can reconstruct input samples with the minimum error (i.e., $\|X - X\|^2 < \rho$, where ρ is a small positive number), we stop the training of networks.

When the trained model is used to test the input data, the final likelihood score is obtained by discriminator D. When the score is greater than the threshold, the input data is normal. Otherwise, the input data is an anomaly. The discrimination mechanism is as follows:

$$OCC_1(\mathrm{X}) = \begin{cases} \text{Target Class} & \text{if } D(R(X)) > \tau \\ \text{Novelty (Outlier)} & \text{otherwise} \end{cases}, \qquad (9)$$

where τ is the detection threshold.



(a)

(b)

Figure 8: Network architecture of R network and D network: (a) R network and (b) D network.

3.3. Algorithm Description. This section will describe the implementation process of the DDoS attack detection method based on all packets in ISP layer in detail. This process mainly includes two stages: training and testing. The detailed process is shown in Algorithm 2.

Algorithm 2 mainly includes the training and testing stage. Lines 01–11 are the train stage, and lines 12–19 are the test stage. In the train stage, line 01 sets the thread pool with m threads. Lines 03–08 obtain the *NCSS*s of the first n normal periods as the training data set. Line 09 reshapes the training data set to a $K/c * K/c$ matrix that is suitable for a convolution network. Through a training AOCC model *epoches* times by train dataset, the optimal model parameters P_best are obtained. In the test stage, the Test$_t$ of the new time period is obtained as lines 13–16, and the test result is obtained according to the trained model.

4. Experimental Analysis

This section mainly describes experiment data sets, evaluation indexes, and detailed performance evaluation of the proposed method. All experiments are carried out in the environment of Inter(R) Core(TM) i7-8565U CPU @ 1.80 GHz 1.99 GHz, RAM 8.0 GB. In the process of experiments, Python is adopted, and the adversarial one-class classification model is implemented in the Keras framework.

4.1. Data Description

4.1.1. MAWILab_BOUN.DDoS Data Set. MAWILab_BOUN.DDS data set is a hybrid data set consisting of MAWI20200501 data set (202005011400.pcap) [31] and BOUN_DDoS data set [32]. In the MAWI20200501 data set, normal network traffic of TCP and UDP protocol is extracted as background traffic. DDoS attack traffic in the BOUN_DDoS data set is extracted as attack traffic. The MAWI20200501 data set is obtained from traffic traces of the MAWI Working Group of the WIDE Project. The traffic traces at the transit link of WIDE to the upstream ISP every day. This link is 1 Gbps with a 150 Mbps committed access rate. MAWI20200501 data set contains 15 minutes of network traffic captured on May 1, 2020, which contains a large number of packets transmitted between tens of

```
Input: Packet set $P$, $K$, $f_1$, $f_2$, $m$, $c$, $n$, epoches
Output: detection result $R$
pool = multiprocessing.Pool(processes = m)
NCSS, R = []
While $i$ in range(0, $n$) do        //The network of first $n$ period is normal.
    SS, $NCSS_i$ = []
    Bucket_List = pool.map_async(ipsketch, $P_i$).get()
    obtain $NCSS_i$ with lines 05–09 of algorithm 1
    NCSS.add($NCSS_i$)
End while
Traindataset = NCSS.reshape(−1, $K/c$, $K/c$,1)
AOCC.train(Traindataset, epoches)
obtain the optimal parameters P_best
While $t$ is continue do
    SS, $NCSS_t$ = []
    Bucket_List = pool.map_async(ipsketch, $P_i$).get()
    obtain $NCSS_t$ with lines (4)–(8) of algorithm 1
    $Test_t$ = $CSS_t$.reshape(−1, $K/c$, $K/c$,1)
    $R_t$ = AOCC.test($Test_t$, P_best)
    R.add($R_t$)
End While
```

Algorithm 2: All-packets-based DDoS attack detection method in ISP layer.

thousands of active hosts (50 million to 200 million packets per 900 seconds). Hence, the MAWI20200501 data set is a representative ISP network traffic data set. BOUN_DDoS data set is recorded on the campus backbone network of Bogazici University with more than 2,000 active hosts. This data set includes normal traffic and attack traffic, in which attack traffic is generated by the random spoofing IP technique of Hping3. The victim is a server connected to the campus backbone router. The attack rate refers to the rate of the number of attack packets to the total packets per unit time in the attack scenario as formula (10) [33]. According to formula (10), the attack rate of this data set is about 1%~10%. The detailed description of the data set is shown in Table 2.

$$\text{Attack}_{\text{rate}} = \frac{P_{\text{attack}}}{P_{\text{total}}} * 100\%, \tag{10}$$

where P_{attack} refers to the number of attack packets and P_{total} refers to the total number of packets.

4.1.2. MAWILab_SDN.DDoS Data Set. MAWILab_SDN.DDoS data set is composed of MAWI20200712 data set (202007121400.pcap) [34] and SDN-DDoS data set [35]. The normal TCP and UDP traffic in the MAWI20200712 data set is extracted as the background traffic, and DDoS attack traffic in the SDN-DDoS data set is extracted as attack traffic. Similar to the MAWI20200501 data set, the MAWI20200712 data set is a 15 minutes network traffic captured from traffic traces of the MAWI Working Group of the WIDE Project on July 12, 2020. SDN-DDoS data sets are ICMP, TCP, and UDP flooding attacks generated by using Scapy and TCPReplay in Mininet Emulator. According to formula (10), the attack rate of the data set is about 4%~15%. The detailed information is shown in Table 3.

4.2. Evaluation Index. To evaluate the detection effect of the proposed method, Acc (accuracy), TPR (true positive rate), FPR (false-positive rate), and FNR (false-negative rate) are adopted in this paper [18]. Acc refers to the proportion of normal and attack behaviors identified correctly in the total traffic behaviors. TPR refers to the proportion of the number of attacks correctly identified as attacks in the total attacks. FPR refers to the proportion of the number of normal behaviors that are mistakenly identified as attacks in the total normal behaviors. FNR measures the proportion of the number of attacks that are wrongly identified as normal behaviors in total attacks. The whole formulas are as follows:

$$\begin{aligned} Acc &= \frac{n_{\text{normal}} + n_{\text{attack}}}{N_{\text{total}}}, \\ TPR &= \frac{n_{\text{attack}}}{N_{\text{attack}}}, \\ FPR &= \frac{n_{\text{normal}\longrightarrow\text{attack}}}{N_{\text{normal}}}, \\ FNR &= \frac{n_{\text{attack}\longrightarrow\text{normal}}}{N_{\text{attack}}}, \end{aligned} \tag{11}$$

where $N_{total} = N_{\text{normal}} + N_{attack}$, N_{total} is the number of total periods, N_{normal} is the number of total normal behaviors, N_{attack} is the number of total attacks, n_{normal} is the number of normal behaviors correctly identified, n_{attack} is the number of attacks correctly identified, $n_{normal\longrightarrow attack}$ is the number of normal behaviors mistakenly identified as attacks, and $n_{\text{attack}\longrightarrow\text{normal}}$ is the number of attacks wrongly identified as normal behaviors.

Table 2: Detailed description of MAWILab_BOUN.DDoS data set.

Data set name	Data set description	Number of total packets	Number of anomaly packets	Duration (s)
MAWILab_BOUN.DDoS	The data set contains normal network traffic and DDoS attack traffic with a rate of 1%~10%.	32, 511, 624	386, 203	900

Table 3: Detailed description of MAWILab_SDN.DDoS data set.

Data set name	Data set description	Number of total packets	Number of anomaly packets	Duration (s)
MAWILab_SDN.DDoS	The data set contains normal network traffic and DDoS attack traffic with a rate of 4%~15%.	22, 015, 542	300, 000	900

4.3. Parameter Set. In this section, the parameters of this paper are set, including the detection period ΔT, the parameters K and c of the square sketch, the parameters of the adversarial one-class classification model, and the threshold τ. Due to the accumulation of the square sketch, ΔT can be set as 1 s, which can be changed by adjusting the number of accumulations of the square sketch. The setting of parameter K depends on the size of the real network. In data set MAWILab_BOUN.DDoS, H is about as 17,000. When $P = 0.0046$, $K = 412$; when $P = 0.00005$, $K = 1{,}304$. In MAWILab_SDN.DDoS, $H = 18{,}000$. When $P = 0.0046$, $K = 424$; when $K = 0.00005$, $K = 1{,}341$. When the conflict rate $P = 0.00005$, K is set as $2^{11} = 2{,}048$ in data sets MAWILab_BOUN.DDoS and MAWILab_SDN.DDoS. For the compressed unit c, if c is too large, the attack information will be covered. If c is too small, it will waste the storage space and is not conducive to the next training of the deep learning model. The threshold τ directly determines the detection result. Therefore, the parameters c and τ are determined by various experiments.

An appropriate adversarial one-class classification model can improve the effect of attack detection. Proper hidden layers can improve the generalization ability of the deep neural network classifier. According to our experience, when the number of input units is not very large, and the number of hidden layers is 3 or 4, the deep neural network has the ability of automatic feature extraction. In the hidden layer, the activation function ReLU can avoid the gradient vanishing problem in the training process. In the output layer, the function sigmoid is generally the activation function. The optimizer RMSprop (lr = 0.001, decay = $1e-8$) and the loss function binary_crossentropy are used to train the AOCC model. For the R network (encoder and decoder) and D neural network, the parameters are set as follows:

(i) Number of hidden layers = {3, 3, 4}

(ii) Activation functions of hidden layers = {LeakyReLU, ReLU, LeakyReLU}

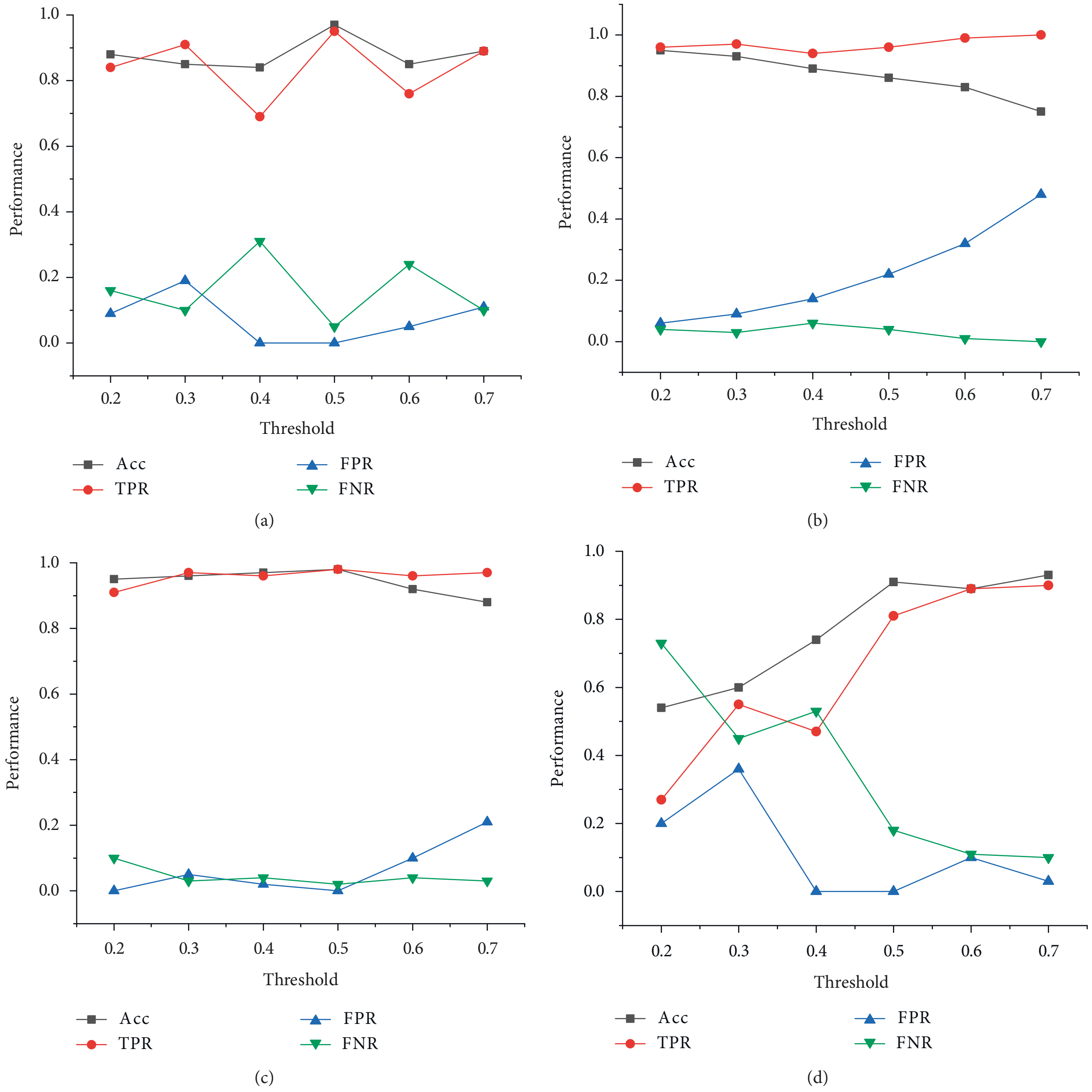
(iii) Activation functions of output layers = {sigmoid, sigmoid}

(iv) Optimizers = {RMSprop (lr = 0.0005, decay = $1e-8$), RMSprop (lr = 0.0005, decay = $1e-8$)}

(v) Loss functions = {binary_crossentropy, binary_crossentropy}

The compression unit c and the threshold τ are set by evaluating the detection effect. Figures 9 and 10 show the detection effect of two data sets under $c = 2^4, 2^5, 2^6, 2^7$ and $\tau = 0.2, 0.3, 0.4, 0.5, 0.6, 0.7$, respectively. Some combinations of the compression unit c and threshold τ produce excellent detection results, for example, when $c = 2^4$ and $\tau = 0.5$ or $c = 2^6$ and $\tau = 0.5$. For data sets MAWILab_BOUN.DDoS and MAWILab_SDN.DDoS, the evaluation results are similar. In this paper, due to the same background traffic of these two data sets, it is very necessary to set the same c and τ because the c value not only determines the detection effect but also determines the size of the square sketch and the training and detection time of the AOCC model, as shown in Figure 11. The size of the square sketch determines the size of data storage. In the training process of the AOCC model, the training time refers to the time consumed by training 703 CSSs with 150 epochs. Test time refers to the testing time of each CSS. With the increase of compression units, data storage, training time, and test time are gradually reduced. From $c = 2^3$ to 2^7, the storage size reduces from 65 KB to 2 KB, the training time reduces from 20,000 s to 400 s, and the test time reduces from 0.03 s to 0.002 s. Therefore, by analyzing the influence of parameters c and τ, $c = 2^6$ and $\tau = 0.5$.

4.4. Analysis of All-Packets Data Mapping Model via Square Sketch. The performance analysis of the all-packets data mapping model via square sketch mainly includes two parts: storage efficiency and time efficiency. In the process of analyzing the storage effectiveness, the storage size of the traditional per-to-per storage mode, square sketch, and compression square sketch is compared as shown in Table 4. It can be seen that the amount of data storage of the original square sketch is the largest. This is because in order to minimize conflicts in data mapping, the size of the square sketch is set to be $2^{11} * 2^{11}$. The size $2^{11} * 2^{11}$ is a very large storage unit. Therefore, the recompression of square sketch is a very important step for decreasing the data storage and time consumption of the detection model. The storage consumption of compressed square sketch is only 5 KB, which is much smaller than that of the traditional per-to-per



(a)

(b)

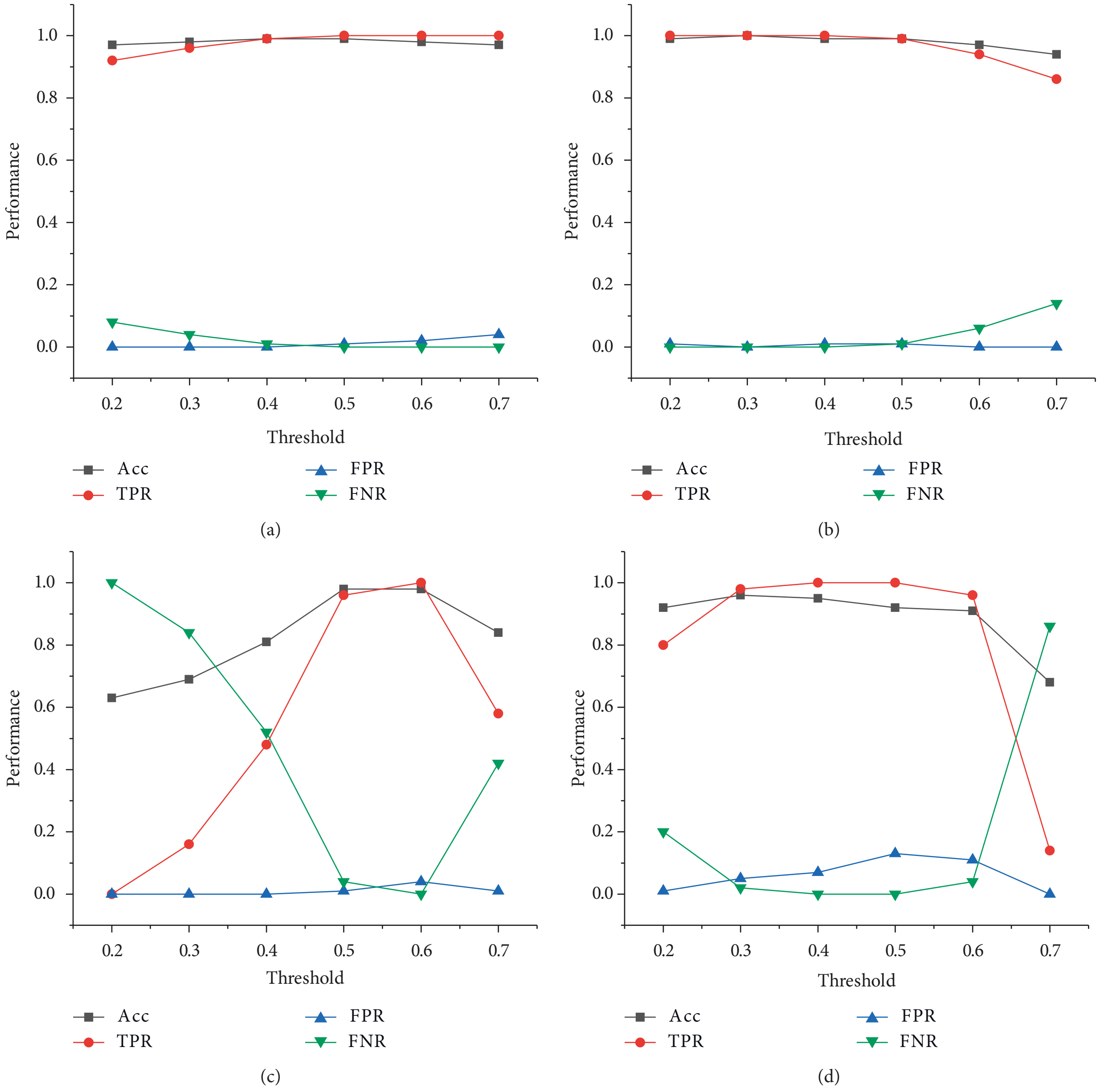
(c)

(d)

Figure 9: Influence of different compressed units on the detection effect in MAWILab_BOUN.DDoS data set: (a) $c = 2_4$, (b) $c = 2_5$, (c) $c = 2_6$, and (d) $c = 2_7$.

storage mode. In addition, in the complex and changeable network environment, once the K and c values of the square sketch are determined, the size of the compressed square sketch would not change with the network environment, which is different from the traditional per-to-per storage mode and conducive to maintaining the stability of the detection model.

Next, we will analyze the time efficiency of the all-packets data mapping model. Compared with the flow-based detection methods, the all-packets-based method reduces the steps of data aggregation from packets to flows. The data aggregation without sampling is a very time-consuming task. Meanwhile, with the increase in network scale, the execution time of aggregation flows is close to exponential growth, which will result in a huge delay in attack detections. In this paper, all packets are mapped into the square sketch. Due to the huge network traffic of the ISP layer, this process will also consume a certain amount of time. In order to achieve less time consumption, the way of multiple threads is executed. Figure 12 shows the time consumption of all packets mapping per unit detection period by the single- and multi-thread ways. From the figure, we can see that in the 1 s detection period, even by the single thread way, the execution time is only 3 s. The execution way of multiple threads enables a faster data mapping process. The execution time is only about 0.7 s in data set MAWILab_SDN.DDoS.



(a)

(b)

(c)

(d)

Figure 10: Influence of different compressed units on the detection effect in MAWILab_SDN.DDoS data set: (a) $c = 2_4$, (b) $c = 2_5$, (c) $c = 2_6$, and (d) $c = 2_7$.

Meanwhile, the execution way of multiple threads will lead to a certain amount of memory and CPU consumption, but the consumption is within an acceptable level. The result gives us great encouragement to realize the distributed data mapping method. Therefore, the all-packets data mapping model via square sketch is an effective data compression and storage model.

4.5. Analysis of the AOCC Model. This section will illustrate the detection performance of the adversarial one-class classification model, mainly including two aspects: detection effect and detection time. In the adversarial one-class classification model, CSS is classified by judging the likelihood score of the discriminator in the AOCC model. The likelihood score results of data sets MAWILab_BOUN.DDoS and MAWILab_SDN.DDoS are shown in Figure 13. In this figure, only a few attack traffic was identified wrongly, and the threshold is 0.5. As can be seen from the figure, the threshold can effectively distinguish attack traffic and normal traffic in these two data sets. This is consistent with the same background network environment of the two data sets. The likelihood score of attack traffic is less than 0.5, and the score of normal traffic is more than 0.5. Although the score difference between attack traffic and normal traffic is not very big, the DDoS attack traffic is still detected by this threshold. Therefore, the AOCC model is an effective detection model. The final detection results are shown in Table 5, which is a very good detection effect. The accuracy is 98%, which illustrates that the AOCC model can

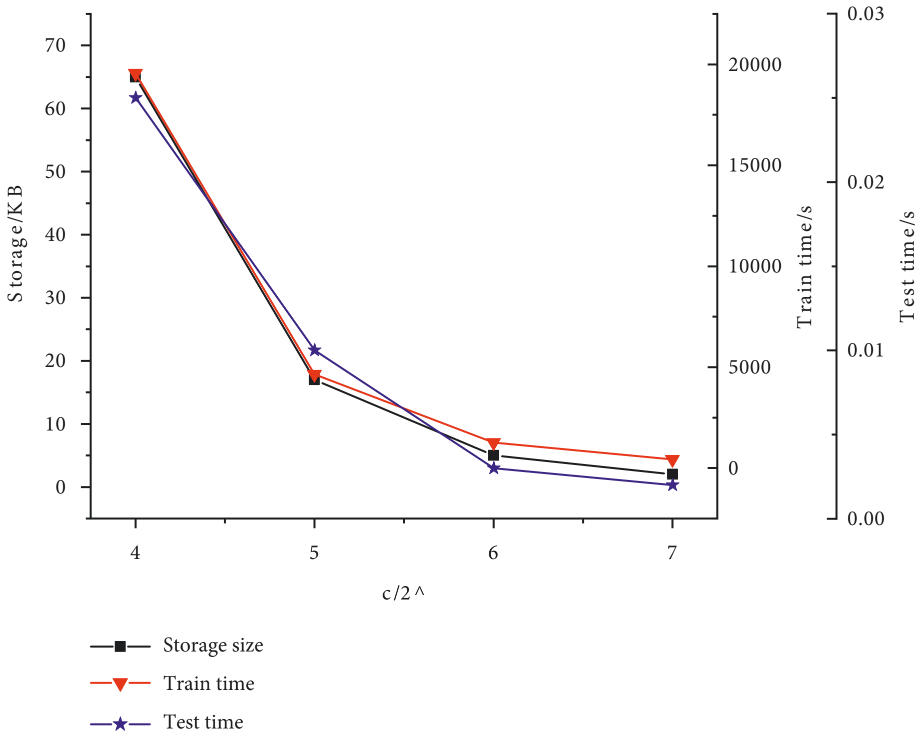


FIGURE 11: Influence of different compressed units on the storage size and execution time in MAWILab_SDN.DDoS data set.

TABLE 4: Comparison of storage size between three storage modes (unit: KB).

	MAWILab_BOUN.DDoS data set	MAWILab_SDN.DDoS data set
Traditional per-to-per storage	1,669	1,102
Square sketch	16,386	16,386
Compress square sketch	5	5

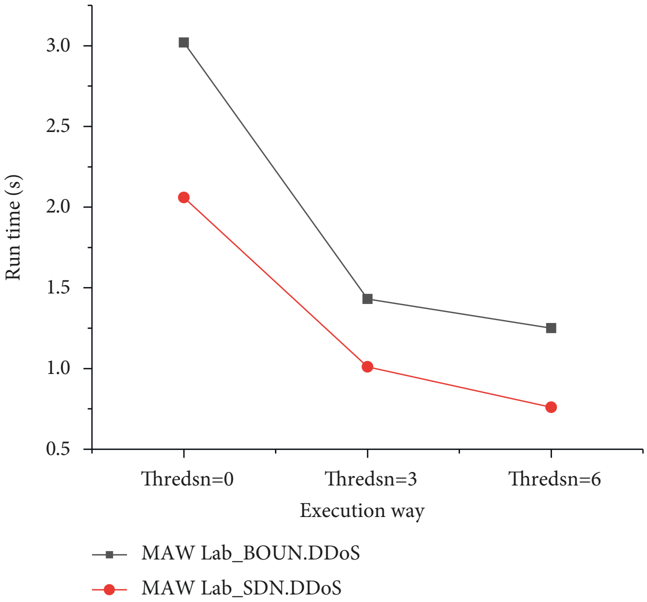


FIGURE 12: Execution time with different threads in the processing of packets mapping.

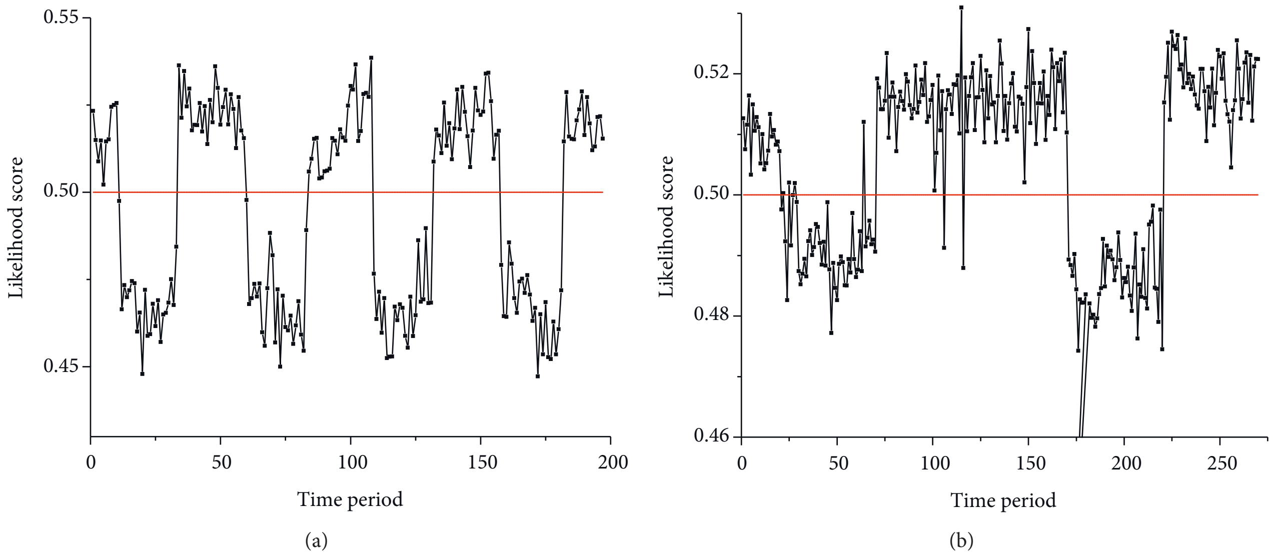


Figure 13: Likelihood score results of adversarial one-class classification model: (a) MAWILab_BOUN.DDoS data set and (b) MAWILab_SDN.DDoS data set.

Table 5: Detailed results of MAWILab_BOUN.DDoS and MAWILab_SDN.DDoS data sets.

	Acc	TPR	FPR	FNR
MAWILab_BOUN.DDoS	0.98	0.98	0	0.02
MAWILab_SDN.DDoS	0.98	0.96	0.01	0.04

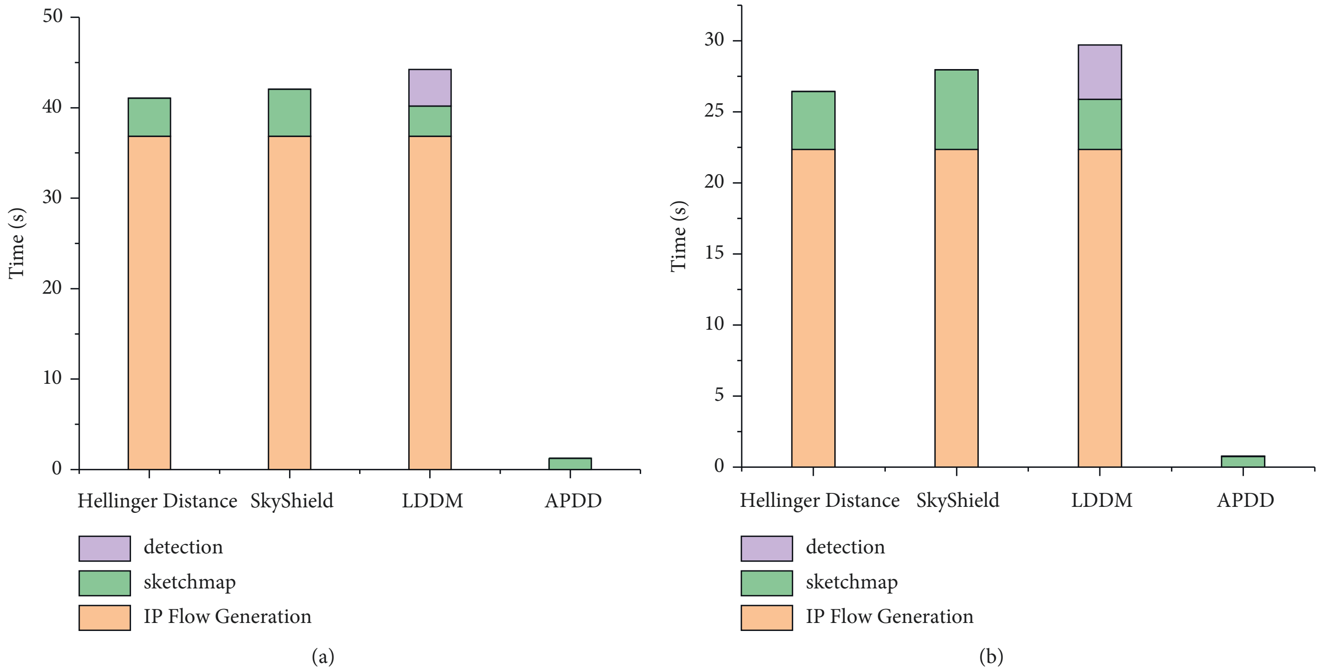


Figure 14: Comparison of running time between APDD and other methods: (a) MAWILab_BOUN.DDoS data set and (b) MAWILab_SDN.DDoS data set.

identify the normal and attack traffic very well. The false-positive rate and false-negative rate are also very low with the highest false-positive rate of 0.01 and the highest false-negative rate of 0.04, which means that the AOCC model will not easily wrongly identify the normal traffic and attack traffic. Therefore, this AOCC model is an effective attack

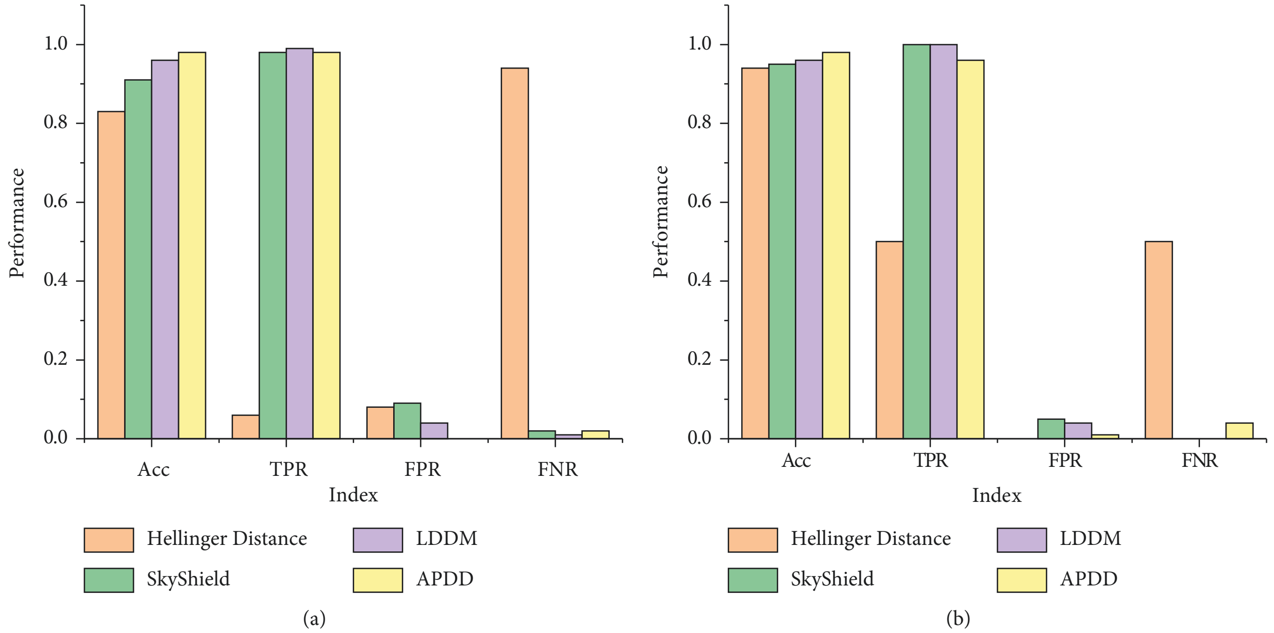


Figure 15: Comparison of detection results between APDD and other methods: (a) MAWILab_BOUN.DDoS data set and (b) MAWILab_SDN.DDoS data set.

detection model. In addition, the detection time of a CSS is only 0.003 s, which means that DDoS attack detection can be quickly realized.

4.6. Comparison with the Existing Methods. Compared with the existing methods, it mainly includes two parts: time effect and detection effect. This paper applies three kinds of comparison methods consisting of Hellinger Distance [36], SkyShield [18], and LDDM [37]. These three methods are based on network flows and use the traditional sketch structure to detect DDoS attacks. The time consumption of our method and the three methods are shown in Figure 14. When our method is executed by multiple threads (such as the number of threads is 6.), the execution time is about 1.25 s and 0.76 s on these two data sets, respectively. The detection time of the AOCC model is only 0.003 s, which can be ignored in practical application. However, for the three flow-based detection methods, the flow generation process takes a lot of time that is far more than the time of sketch mapping and attack detection. Therefore, it is an important strategy to build the model based on the original packets, which plays an extremely positive role in realizing real-time DDoS attack detection. Meanwhile, the experiment results show that our method has an excellent time performance.

In order to further analyze the detection effect, the detection effect of our method is compared with the above three flow-based detection methods. The detection results are shown in Figure 15. From the figure, our method has an excellent accuracy that is close to 1 and higher than the other three methods. In other words, this method has a good detection ability for distinguishing normal and abnormal CSSs. Meanwhile, this method has a good TPR, which is basically equal to the best TPR. The FPR and FNR are also better than or equal to other methods. Skyshield and LDDM methods have slightly lower detection accuracy than our proposed method and a higher false-positive rate. The Hellinger distance [36] method has the lowest detection effect, which is due to the inapplicability of the Hellinger distance method and the threshold method in MAWILab_BOUN.DDoS and MAWILab_SDN.DDoS data sets. Therefore, this method has an excellent detection ability and can detect almost all attacks. In conclusion, combined with the time efficiency and detection performance of our method, our method has better detection effect and time efficiency than the existing methods, which can detect multi-rate DDoS attacks and is completely suitable for the ISP layer.

5. Conclusion

In order to realize the effective DDoS attack detection in the ISP layer, this paper proposes a new all-packets-based DDoS attack detection method. Firstly, a novel probabilistic data structure, square sketch, is designed. The characteristics of parallelization, accumulation, and recompression are analyzed. According to the source IP and destination IP information, all packets in a unit period are hashed to the square sketch. The mapped square sketch is further recompressed to obtain a new compressed square sketch with a small size. This process completely omits the generation of network flows in the traditional DDoS attack detection methods, which effectively shortens the intermediate steps of network traffic and the time delay of attack detection. The experimental results show that the execution time of all packets mapping process via multiple threads is less than the unit detection period even in the detection period of 1 s. Next, only utilizing the historical normal compressed square sketches, the DDoS attack detection model based on the adversarial one-class classification model is obtained. According to multiple training, the optimal attack detection model is realized. The compressed square sketch of a new period is tested to get the detection results. Experimental results show that our attack detection model can get an effective detection effect only based on normal traffic, which is higher than or equal to the existing DDoS attack detection methods. At the same time, the detection time of the proposed APDD method is much better than the existing detection methods.

Data Availability

Data are available at http://www.fukuda-lab.org/mawilab/v1.1/2020/05/01/20200501.html, https://ieee-dataport.org/open-access/bo%C4%9Fazi%C3%A7i-university-ddos-dataset, http://www.fukuda-lab.org/mawilab/v1.1/2020/07/21/20200721.html, and https://data.mendeley.com/datasets/hkjbp67rsc/1.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61772449, 61802332, and 61807028), the Natural Science Foundation of Hebei Province P. R. China under Grant no. F2019203120, Langfang Science and Technology Research and Development Plan under Grant no. 2021011038, the Scientific Research Project of Colleges and Universities in Hebei Province under Grant no. QN2021144, and Science and Technology Project of Hebei Education Department (BJK2022029, QN2021145).

References

[1] K. Doshi, Y. Yilmaz, and S. Uludag, "Timely detection and mitigation of stealthy DDoS attacks via IoT networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 99, pp. 2164–2176, 2021.

[2] S. Shamshirband, "Computational intelligence intrusion detection techniques in mobile cloud computing environments: review, taxonomy, and open research issues," *Journal of Information Security and Applications*, vol. 55, p. 102582, 2020.

[3] A. Network, "Worldwide infrastructure security report," 2016, http://www.arbornetworks.com/images/documents/wisr2016enweb.pdf.

[4] S. Alzahrani and L. Hong, "Generation of DDoS attack dataset for effective IDS development and evaluation," *Journal of Information Security*, vol. 09, no. 4, pp. 225–241, 2018.

[5] X. Jing, Z. Yan, X. Jiang, and W. Pedrycz, "Network traffic fusion and analysis against DDoS flooding attacks with a novel reversible sketch," *Information Fusion*, vol. 51, pp. 100–113, 2018.

[6] S. İ. N. A. N. Toklu and M. Şimşek, "Two-layer approach for mixed high-rate and low-rate distributed denial of service (DDoS) attack detection and filterin," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 7923–7931, 2018.

[7] M. E. Ahmed, S. Ullah, and H. Kim, "Statistical application fingerprinting for DDoS attack mitigation," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1471–1484, 2018.

[8] D. K. Sharma, T. Dhankhar, G. Agrawal et al., "Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks," *Ad Hoc Networks*, vol. 121, p. 102603, 2021.

[9] Mawilab, "Mawi Working Group Traffic Archive (2014)," 2014, http://mawi.wide.ad.jp/mawi/ditl/ditl2014/201410022315.html.

[10] A. A. Amaral, "Deep IP flow inspection to detect beyond network anomalies," *Computer Communications*, vol. 98, pp. 80–96, 2017.

[11] R.-H. Hwang, M.-C. Peng, V.-L. Nguyen, and Y.-L. Chang, "An lstm-based deep learning approach for classifying malicious traffic at the packet level," *Applied Sciences*, vol. 9, no. 16, p. 3414, 2019.

[12] M. Kallitsis, S. A. Stoev, S. Bhattacharya, and G. Michailidis, "AMON: an open source architecture for online monitoring, statistical analysis, and forensics of multi-gigabit streams," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 6, pp. 1834–1848, 2016.

[13] A. Abou Daya, M. A. Salahuddin, N. Limam, and R. Boutaba, "A graph-based machine learning approach for bot detection," in *Proceedings of the 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, Arlington, VA, USA, April 2019.

[14] J. Wang and I. C. Paschalidis, "Botnet detection based on anomaly and community detection," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 392–404, 2017.

[15] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting http-based application layer dos attacks on web servers in the presence of sampling," *Computer Networks*, vol. 121, pp. 25–36, 2017.

[16] C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Combining sketches and wavelet analysis for multi time-scale network anomaly detection," *Computers & Security*, vol. 30, no. 8, pp. 692–704, 2011.

[17] Q. Huang and P. P. Lee, "A hybrid local and distributed sketching design for accurate and scalable heavy key detection in network data streams," *Computer Networks*, vol. 91, pp. 298–315, 2015.

[18] C. Wang, T. T. N. Miu, X. Luo, and J. Wang, "Skyshield: a sketch-based defense system against application layer DDoS attacks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, pp. 559–573, 2018.

[19] X. Jing, J. Zhao, Q. Zheng, Z. Yan, and W. Pedrycz, "A reversible sketch-based method for detecting and mitigating amplification attacks," *Journal of Network and Computer Applications*, vol. 142, no. 15, pp. 15–24, 2019.

[20] Y.-S. Jeong, I.-H. Kang, M.-K. Jeong, and D. Kong, "A new feature selection method for one-class classification problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1500–1509, 2012.

[21] V. H. Bezerra, V. G. T. da Costa, S. Barbon Junior, R. S. Miani, and B. B. Zarpelão, "Iotds: a one-class classification approach to detect botnets in internet of things devices," *Sensors*, vol. 19, no. 14, p. 3188, 2019.

[22] Q. Wang, J. Huang, Y. Feng, Z. Luo, C. Fan, and X. Liang, "A comprehensive revisit to one class classification," in *Proceedings of the 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, Beijing, China, March 2017.

[23] B. B. Gupta, M. Misra, and R. C. Joshi, "An isp level solution to combat DDoS attacks using combined statistical based approach," *Journal of Information Assurance and Security*, vol. 3, pp. 102–110, 2008.

[24] N. Hinze, M. Nawrocki, M. Jonker, A. Dainotti, T. C. Schmidt, and M. Wählisch, "On the potential of bgp flowspec for DDoS mitigation at two sources: ISP and IXP," in *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*, Budapest, Hungary, August 2018.

[25] Z. Liu, Y. Cao, M. Zhu, and W. Ge, "Umbrella: Enabling Isps to Offer Readily Deployable and Privacy-Preserving DDoS Prevention Services," *Cryptography and Security*, vol. 14, pp. 1098–1108, 2019.

[26] C. Gong, T. Shi, M. Mu, L. Zhao, A. Gani, and H. Qi, "An improved quantum genetic algorithms and application for DDoS attack detection," in *Proceedings of the ISPA/BDCloud/SocialCom/SustainCom*, Xiamen, China, December 2019.

[27] I. Ko, D. Chambers, and E. Barrett, "Feature dynamic deep learning approach for DDoS mitigation within the isp domain," *International Journal of Information Security*, vol. 19, pp. 53–70, 2020.

[28] I. Ko, D. Chambers, and E. Barrett, "Self-supervised network traffic management for DDoS mitigation within the isp domain," *Future Generation Computer Systems*, vol. 112, pp. 524–533, 2020.

[29] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision Pattern Recognition*, Salt Lake City, Utah, June 2018.

[30] X. Zhou, H. Chen, L. Yan, Y. Xu, and X. He, "Avae: adversarial neural network for self-representation in one class classification," in *Proceedings of the 2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, Chengdu, China, December 2019.

[31] MAWILab, "Fukuda-lab," 2020, http://www.fukuda-lab.org/mawilab/v1.1/2020/05/01/20200501.html.

[32] D. Erhan, "Boğaziçi university ddos dataset," 2020, https://ieee-dataport.org/open-access/bo%C4%9Fazi%C3%A7i-university-ddos-dataset.

[33] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 2018.

[34] MAWILab, "Fukuda-lab," 2020, http://www.fukuda-lab.org/mawilab/v1.1/2020/07/21/20200721.html.

[35] O. G. Housman, H. Isnaini, and F. Sumadi, "SDN-DDOS (ICMP,TCP,UDP)," 2020, https://data.mendeley.com/datasets/hkjbp67rsc/1.

[36] J. Tang, Y. Cheng, and C. Zhou, "Sketch-based sip flooding detection using Hellinger distance," in *Proceedings of the 2009 IEEE Conference on Global Telecommunications*, Honolulu, HI, USA, November 2009.

[37] X. Liu, J. Ren, H. He, Q. Wang, and C. Song, "Low-rate DDoS attacks detection method using data compression and behavior divergence measurement," *Computers & Security*, vol. 100, p. 102107, 2021.