Future Generation Computer Systems 132 (2022) 67–79

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A feature selection-based method for DDoS attack flow classification

Lu Zhou [a,*], Ye Zhu [a], Tianrui Zong [b], Yong Xiang [a]

[a] School of Information Technology, Deakin University, Victoria, 3125, Australia
[b] ZHONGTUKEXIN CO., LTD., Beijing 100020, China

ARTICLE INFO

Article history:
Received 15 September 2021
Received in revised form 30 December 2021
Accepted 10 February 2022
Available online 16 February 2022

Keywords:
DDoS attacks
Flow classification
Machine learning
Feature selection

ABSTRACT

Distributed Denial of Service (DDoS) attacks still be a great threat to the availability of online servers. To defend against attacks, the challenge is not only detecting DDoS attacks as they occur but also identifying, and thus blocking the attack flows. However, existing classification methods cannot accurately and efficiently differentiate between attack flows and benign flows. In this paper, we propose a DDoS attack flow classification system, named SAFE, to accurately and quickly identify attack flows in network layer. First, SAFE chooses the optimal features by removing the redundant features and selecting the most informative features. Second, a threshold tuning method is proposed to identify the best threshold for each feature. Finally, an aggregated feature-based linear classifier is proposed to weight the selected features for classification. Since the proposed method monitors the flows in network layer, it can detect the traditional DDoS attack flows as well as the attack flows launched by Internet of Thing (IoT) devices. Comprehensive experiments are carried out on one IoT and two sophisticated DDoS attacks to evaluate the classification performance of the proposed method. The comparison results show that SAFE can achieve better classification performance than the state-of-the-art methods in terms of classification accuracy and efficiency.

1. Introduction

Distributed Denial of Service (DDoS) attacks have been a great threat to the availability of online service. A successful DDoS attack aims to exhaust the resources of an online service using malicious traffic form multiple sources so that normal users cannot gain access. According to an annual report, the size of DDoS attacks keeps growing at an alarming pace in recent years, in which 400 Gbps attacks are now a matter of routine and the largest reported attack has reached 1.7 Tbps targeted a North American service provider in 2018 [1]. Moreover, the complexity of attack types, which are volumetric, TCP state exhaustion, and application-layer, exploring different vulnerabilities of a victim makes it even more difficult to defend [2].

To defend against DDoS attacks, various defenses have been proposed [3–9]. Most of these methods focus on detecting attacks (e.g., whether a DDoS attack is ongoing). The detection systems provide the foundation for DDoS attack defense. But the victim protection is limited since they cannot stop the attacks. To inherently stop DDoS attacks, the key challenge is to distinguish the attack flows from the benign flows and, therefore, block the attackers' flows.

Some methods have been proposed to identify attack flows [10,11] that can be classified into two categories: payload inspection-based classification and machine learning-based classification. Payload inspection depends on inspecting the packet payloads (e.g., HTTP message content [10]) to obtain the attack characteristics. However, the classification performance is low for attacks targeting protocol vulnerabilities, such as SYN flooding and ICMP flooding, since the packet payloads contain limited information. Moreover, the method involves privacy issues since it needs to inspect packet content, i.e., users' application data. The existing machine learning-based method (e.g., the long short-term memory (LSTM) network [11]) requires a large number of features to train the enormous number of parameters, which is time consuming. Even when the model has been trained offline, it is still expensive to fit the enormous parameters for every flow sample in the identification phase, especially when a victim is suffering attacks and the computational resources are limited.

In this paper, we go beyond detecting whether a DDoS attack is ongoing; we attempt to identify the attack flows in network layer when an attacker performs an attack. Therefore, we propose a DDo**S** **A**ttack **F**lows id**e**ntification (SAFE) system, which can effectively and efficiently identify DDoS attack flows. Compared to previous work, the proposed method provides more precise classification results and significantly reduces the time consumption for attack flow classification. Moreover, the method does not rely on payload inspection to obtain features and thus protect the users' privacy.

To achieve this goal, we need to address the following challenges. The first challenge is to select the informative features

* Corresponding author.
E-mail address: zhoulu000@gmail.com (L. Zhou).

https://doi.org/10.1016/j.future.2022.02.006
0167-739X/

that can distinguish the attack flows from benign flows. To address this problem, SAFE first extracts 21 features in which the attack flows present abnormalities from three categories: application, volume, and TCP state. Then, a correlation-based method is deployed to measure the linear correlation between the features and thus removes redundant features. Finally, SAFE ranks the features according to importance and selects the most informative features for classification.

The second challenge is to find the optimal thresholds for the selected features. To address this challenge, we design a threshold tuning method to search for the best threshold for each feature. The selected thresholds can maximize the classification performance according to the input features.

The third challenge is that the flow classification method should be lightweight to reduce the time consumption in the classification phase. To do this, we propose an aggregated feature-based linear classifier. This is achieved by combining the outputs from the selected features and the feature weights. Therefore, there is no need to train the enormous parameters, such as in the case of LSTM method [11].

To summarize, our contributions are listed as follows:

- We propose a feature selection method to select features for DDoS attack flow classification. The method can reduce the number of features by selecting the independent and most relevant features for classification.
- We design the SAFE architecture to identify both Internet of Thing (IoT) and sophisticated DDoS attack flows. First, SAFE deploys a threshold tuning method to find the optimal thresholds for the selected features. Second, SAFE utilizes an aggregated feature-based classification method by combining the outputs of the selected features and the feature weights.
- We present a comprehensive evaluation of SAFE on one IoT and two sophisticated DDoS attacks. The empirical study shows that SAFE can effectively identify attack flows within a few features and outperforms the compared methods in terms of effectiveness and efficiency.

The rest of the paper is organized as follows: Section 2 reviews related work in DDoS attack detection and attack flow classification. A feature selection-based classification approach is proposed in Section 3. Section 4 presents the performance analysis and comparison experiments. Finally, the paper is concluded in Section 5.

2. Related work

2.1. DDoS attack detection

DDoS attack detection, such as anomaly detection and machine learning-based classification, has been widely studied. Anomaly detection aims to measure abnormal deviations in certain feature, and a DDoS attack can be detected if the deviation exceeds the threshold [12]. In [13], the authors proposed a generalized entropy-based method to detect low-rate DDoS attacks. The method is based on the probability distribution difference between normal traffic and low-rate attack traffic, and traffic is classified as a low-rate attack when the generalized entropy drops. The authors in [7] proposed RADAR, an adaptive correlation analysis-based method, to detect and throttle SYN flooding attacks. The method is based on analyzing the abnormal SYN to ACK packet ratio, and a SYN flooding attack is detected when the SYN to ACK packet ratio exceeds the threshold.

Machine learning methods have been proposed to detect DDoS attacks. The authors in [9] proposed a learning-driven detection method to detect and mitigate DDoS in IoT. They used a semi-supervised machine-learning algorithm to detect and mitigate DDoS attack traffic. In [14], the authors proposed a deep learning method to detect DDoS attacks. They designed a bidirectional recurrent deep neural network to learn patterns and identify DDoS attacks. The authors in [15] also proposed a deep learning-based model, a stacked AutoEncoder (SAE) model, to effectively detect DDoS attacks. They utilized adaptive polling sampling to process the traffic samples and classify the benign and attack traffic using the SAE model. The method in [16] evaluated the effectiveness of several classification algorithms such as Gated Recurrent Units (GRU), Recurrent Neural Networks (RNN), Naive Bayes (NB) for DDoS attacks detection and concluded that GRU achieves the best detection performance. The authors in [17] proposed a cognitive-inspired computing-based DDoS attacks detection mechanism. The mechanism selected a few features and utilized SVM to classify normal and attack traffic. In [18], the authors proposed a method based on K-means and fast K-nearest neighbors to detect DDoS attacks in software-defined networking (SDN). However, these methods were designed to detect whether a DDoS attack is ongoing, therefore, they cannot identify the attackers, i.e., they cannot distinguish the attack flows from normal flows.

2.2. DDoS attack flow classification

Some methods on attack flow identification have been studied in a recent review [19]. In [20], a decision-tree-based architecture was proposed to distinguish attack flows and benign flows, and then the features with low variances were selected and fed into a decision tree C4.5 algorithm. In [21], an ensemble framework was proposed to select features for DDoS attack flow classification, where seven feature selection methods were used to individually select features and the majority voting technique was applied to choose the final 11 features. The method in [22] chose 5 methods to select features and concluded that the selected 10 features with Naive Bayes can achieve the best performance. In [23], the authors proposed a detection framework and selected 16 features to detect DDoS attacks. However, the feature selection algorithms adopted by the methods in [20–23] are not designed for the imbalanced data distribution that is the inherent character in real DDoS attacks. As a result, the features extracted by the methods in [20–23] are not tailored for real DDoS attacks detection.

In [10], based on the theory that users belonging to a botnet are expected to exhibit a smaller degree of message innovation than normal users, the authors proposed a message innovation rate-based method to identify DDoS botnets. Two indicators, the transmission rate and the number of distinct messages per unit time, were employed in the classification. However, the method heavily relies on inspecting the packet's payload, e.g., HTTP message content, to count the number of distinct messages, which is a privacy concern. Moreover, the classification accuracy is limited since it only focuses on attacks that utilize HTTP packets. The authors in [11] proposed a deep learning-based model, a LSTM network, to identify attack flows. They used 40 flow features as input and fed them into an LSTM model. The detection accuracy is high but it is hard to understand and interpret why it performs well. Besides, the method is highly time consuming since thousands of parameters are needed to fit for each flow even after the model has been trained. The problem of effectively and efficiently identifying the DDoS attack flows remains to be solved.

In the proposed method, we select our features using the feature importance-based area under the receiver operating characteristics curve (AUC) that is specifically designed for the imbalanced data distribution. Therefore, the features we extract are more efficient in detecting real DDoS attacks. Besides, we propose a novel aggregated feature-based linear algorithm to weight the

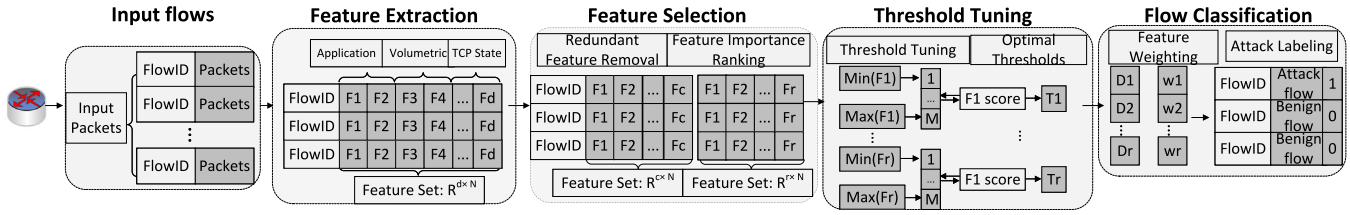


Fig. 1. The overview of the flow classification. Phase 1 classifies the input packets into flows. Phase 2 computes and extracts features for each flow. Phase 3 selects and ranks features. Phase 4 chooses the optimal thresholds for the selected features. Phase 5 weights the feature outputs and labels attack flows.

Table 1
Notations used in this paper.

Notation	Meaning
X	The flow samples
N	The number of flow samples
d	The number of high-dimension feature
F	The original feature set in the classification
F_i	The ith feature in the classification
f_i^j	The jth feature value for the ith feature
δ	The threshold of Pearson correlation coefficient
r	The number of selected features
$r_{(i,j)}$	the PCC value between the ith and jth feature
$AUC(f_i)$	The AUC score of the ith feature
$FA(f_i)$	The feature importance-based AUC score of the ith feature
F'	The selected feature set in the feature selection algorithm
M	The number of thresholds for feature f
$T(i)$	The ith threshold value for feature f
α	The optimal threshold for feature f
y	The true label of the flow samples
y_{pred}	The predicted label in the feature threshold tuning algorithm
$D[i][j]$	The predicted output of the ith feature for the jth flow
W	The weight of the selected features in the classification
ε	The bias of the selected features in the classification
L_j	The output of the jth flow in the classification
y_j'	The final predicted label for the jth flow

Table 2
Details of the extracted features.

Application	Description
F1. src port	Source port number
F2. dst port	Destination port number
F3. protocol	Type of protocol
Volumetric	
F4. flow duration	Duration of the flow
F5. tot fwd pkts	Total number of forward packets
F6. tot bwd pkts	Total number of back-forward packets
F7. tot len bwd pkts	Total length of back-forward packets
F8. fwd pkts len min	The min length of the forward packets
F9. fwd pkts len std	The length variance of the forward packets
F10. bwd pkts len max	The max length of the back-forward packets
F11. flow bytes rate	Number of bytes of the flow per second
F12. bwd IAT tot	Total time interval of the back-forward packets
F13. bwd IAT mean	Mean time interval of the back-forward packets
F14. fwd psh num	Number of PSH flag packets in forward packets
F15. bwd pkts rate	Number of back-forward packets per second
F16. fwd ent int	Entropy of time interval of the forward packets
TCP state	
F17. fwd TCP num	Number of TCP of the forward packets
F18. fwd ACK num	Number of ACK flags of the forward packets
F19. fwd max ACK int	Max ACK flag interval of the forward packets
F20. fwd SYN num	Number of SYN flags of the forward packets
F21. fwd SYN rate	Rate of SYN flags of the forward packets

selected features and the thresholds, which further enhances the detection accuracy with limited features. As a result, compared to [11] and [20–23], the proposed method can significantly reduce the number of features used in detection and thus greatly improve the detection efficiency. In terms of privacy protection, different from [10], the proposed method does not inspect the packet's payload and consequently can protect the user's privacy. Moreover, instead of only detecting HTTP packets [10], the proposed method monitors all types of packets to improve detection accuracy.

3. The proposed method

To defend against DDoS attacks, the problem is to identify the attack flows and block the flows from the attackers. A flow can be represented as several packets with the same five-tuple information (source IP address, destination IP address, source port number, destination port number, protocol) [24], and an attacker flow is a flow in which the source IP address belongs to the attacker. Assume that there are N flow samples and y classes. Let the flow sample be $X = \{F_1, F_2, \ldots, F_N\} \in R^{d\times N}$, where F_i is the ith flow, d is the number of original features and N is the number of flows. The true label of flow F_i is denoted as y_i, $y_i = \{0, 1\}$. Our aim is to build an end-to-end method to predict a label $y_{pred}(i)$ that is exactly the true label y_i. All of the symbols used in this paper are listed in Table 1.

This section presents a feature selection-based flow classification scheme to classify attacks and benign flows, which can significantly improve classification accuracy and efficiency. Fig. 1 shows the SAFE operation phases: First, SAFE classifies the incoming packets into a flow table according to the five-tuple information from the packet header. It then computes and extracts features for each individual flow's packets 3.1. This is followed by removing the redundant features and selecting the most informative features 3.2. It then chooses the optimal thresholds for the selected features 3.3. Finally, SAFE combines the outputs from the selected features with the feature weights and verifies the attack flow labels 3.4.

3.1. Feature extraction

Feature extraction is performed to obtain the potential effective features from the packets of each flow. To preserve as many distinctive features as possible, the feature extraction focuses on the features of the attack flows that present any deviations from the benign flows. Since the aim of a DDoS attacker is to exhaust the resources of a victim and the resource exhaustion strategies are mainly separated into three categories: volumetric-based, TCP state exhaustion-based, and application layer-based [2], we extract 21 unidirectional statistical features from these categories. It is reasonable to assume that we have packets of both attack and benign flows so we can extract the features of the flows. The details are presented in Table 2 and all features are normalized using the min–max normalization [25].

3.1.1. Application layer-based

Application-layer DDoS attacks explore the vulnerability caused by the specific open service ports to launch the attack (e.g., HTTP flooding) [26]. An attacker optionally targets these open ports (e.g., 80 of HTTP) and launches flooding attacks by

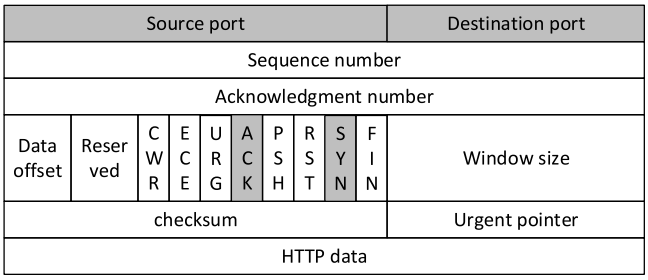


Fig. 2. An example of the fields for feature extraction in the TCP packet header.

building a standard connection through firewalls. Therefore, the features, source port and destination port monitor the flow with a specific port number. In addition, we also observe that the main protocols utilized by the attackers are HTTP, TCP, UDP, and ICMP [8]; the feature, *protocol*, records the protocol number of the flows.

3.1.2. Volumetric-based

Another indicator of suspicious activity in DDoS attacks is the volume of the flows since 42% of the attacks are still volumetric attacks [1], and these attacks present a large volume of network layer packets. We measure the statistical characteristics of the flows and classify them into three categories: packet number-based, packet length-based, and packet time interval-based. The packet number-based features, such as F5 *tot fwd pkts*, count the packets in the flows. The packet length-based features measure the abnormal packet length distribution, such as F8 *fwd pkts len min* and F9 *fwd pkts len std*, and the packet time interval-based features measure the packet rate of the flows, such as F12 *bwd IAT tot* and F13 *bwd IAT mean*.

3.1.3. TCP state exhaustion-based

TCP state-exhaustion attacks attempt to exhaust TCP state tables in the process of building three-way handshaking with bogus connections and thus disrupt connections from benign users. For example, a SYN flooding attack opens massive half-open TCP connections by deliberately sending SYN packets and keeping the corresponding final ACK to exhaust the SYN-queue resources [27]. Therefore, we extract 5 statistical features, such as F20 *fwd SYN num* and F18 *fwd ACK num*, to measure the abnormal behavior related to TCP state-exhaustion attacks.

Instead of inspecting the packet payload to obtain the features [10], the chosen features in this paper are extracted from the packet headers fields (e.g., F1 *src port* and F2 *dst port*) and the statistical information of the flow, such as F5 *tot fwd pkts* and F17 *fwd TCP num*. An example of feature extraction in the TCP packet header fields is shown in Fig. 2. All of these features do not rely on inspection of the packet payload; therefore, user privacy is protected.

Remark 1. Since the proposed method focuses on flow features, in this paper, we have considered up to 120 flow features including the flow features from cicflowmeter [28]. However, due to the length limitation, it is not feasible to visually display the feature selection processing for all 120 features in the manuscript. Therefore, for demonstration purposes, we filtered out 99 features based on their feature importance-based AUC (FA) score rankings, which is introduced later in Section 3.2.2, beforehand and only used the aforementioned top 21 features to illustrate the feature selection process. Nevertheless, we have provided all of these 120 features in Table A.1 in Appendix.

3.2. Feature selection

Although some of these features are relevant to determining the attack flow classification, some features may have little or no effect on the classification result, which increases time and process costs. To achieve efficient and accurate classification, we select the best features that can distinguish the attack flows and benign flows. We use feature selection methods that do not change the original features but merely select a subset of features that are most useful. Let $\{F_1, F_2, \ldots, F_d\}$ be d features of X, where d is the number of high-dimension features. To select the most discriminative flow features for attack flow recognition, we assume there is a method that makes high-dimension d to low-dimension r ($r < d$). Therefore, the feature selection process needs to meet the following two principles: (i) the redundant features are removed and (ii) the most relevant features are selected.

3.2.1. Redundant features removal

Redundant features are the features in which the values are highly correlated to others', and it is sufficient to select only one of them. To remove the redundant features, we use the Pearson correlation coefficient (PCC) to evaluate the dependency between the features. Let $F_i = \{F_i^1, F_i^2, \ldots, F_i^N\}$ be the ith feature vector from N samples, where $F_i \in \{F_1, F_2, \ldots, F_d\}$, and (F_i, F_j) $(1 \leqslant i \leqslant j \leqslant d)$ be the feature pairs of the ith and jth features from the flow samples. The PCC of the two feature is defined as:

$$r_{(i,j)} = \frac{\sum_{k=1}^{N}(F_i^k - \bar{F}_i)(F_j^k - \bar{F}_j)}{\sqrt{\sum_{k=1}^{N}(F_i^k - \bar{F}_i)^2 \sum_{k=1}^{N}(F_j^k - \bar{F}_j)^2}}. \tag{1}$$

where $r_{(i,j)} \in [-1, 1]$. When $r_{(i,j)} \to -1$, features F_i and F_j are negatively linearly related; when $r_{(i,j)} \to 1$, the two features are positively linearly related. When the two features are irrelevant, we have $r_{(i,j)} = 0$. We evaluate the dependency between features F_i and F_j according to PCC, and features F_i and F_j are redundant if $|r_{(i,j)}|$ reaches or exceeds a threshold δ. Thus, we remove the feature F_j, and the feature F_i will be included in the candidate feature set.

3.2.2. Relevant features selection

To select the relevant features, we employ the AUC [29] to weigh the importance of individual features in the classification. The AUC evaluates the probability that a randomly chosen attack sample has a higher modeled probability than a randomly benign sample [29]. We use AUC to rank and select features, especially for binary classification with imbalanced data distribution [30].

For a given feature $f = \{(\mathbf{x}_i, y_i) \in R \times \{-1, +1\} | i \in [n]\}$, where $[n] = \{1, 2, \ldots, n\}$. The attack flow sample set is defined as $f^+ = \{(\mathbf{x}_i^+, +1) | i \in [n_+]\}$, and the benign flow sample set is $f^- = \{(\mathbf{x}_j^-, -1) | j \in [n_-]\}$, where n_+ and n_- are the numbers of attack and benign flows, respectively. The AUC of feature f is:

$$AUC(f) = \sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \frac{G[f(\mathbf{x}_i^+) > f(\mathbf{x}_j^-)] + \frac{1}{2}G[f(\mathbf{x}_i^+) = f(\mathbf{x}_j^-)]}{n_+ n_-} \tag{2}$$

where $AUC(f) \in [0, 1]$. $G[\pi]$ is the indicator function and $G[\pi] = 1$ when π holds, while $G[\pi] = 0$ otherwise. When $AUC(f) = 0.5$, feature f is an irrelevant feature to classification; when $AUC(f) \to 1$, feature f is highly related to the attack class, while it is related to the benign class when $AUC(f) \to 0$ [30].

The AUC score $AUC(f_i)$ can be twofold. First, it can identify the relationship of the feature value between the normal flow and attack flow. If $AUC(f_i) > 0.5$, it indicates that the values of normal flows are generally smaller than those of attack flows in feature f_i. If $AUC(f_i) < 0.5$, the values of normal flows are greater

than those of attack flows in feature f_i. Therefore, we can label the flow samples according to $AUC(f_i)$ (shown in Section 3.4). Second, it evaluates the feature importance and selects the most informative features. According to the AUC score definition, the feature importance of f_i can be evaluated as the distance between the AUC score $AUC(f_i)$ and 0.5; therefore, we use the FA score to evaluate feature importance, which is defined as:

$$FA(f_i) = \begin{cases} AUC(f_i), & \text{if } AUC(f_i) \geqslant 0.5 \\ 1 - AUC(f_i), & \text{if } AUC(f_i) < 0.5 \end{cases} \tag{3}$$

where $FA(f_i) \in [0.5, 1]$. A feature with a higher FA score implies that a linear classification algorithm can obtain a good classification result by setting a threshold or decision boundary on that feature [31]. Thus, the features can be ranked in descending order according to the FA score and we can select the first r features as the ultimate result.

Therefore, a feature f_i is selected if (i) the PCC values to other features do not exceed a threshold δ; (ii) it is one of the top r features according to the FA score. Algorithm 1 shows the feature selection details. Its computational complexity is $O(d^2)$, where d is the number of original features. Lines 1–10 in Algorithm 1 describe how it computes the PCC of features f_i and f_j and removes the feature f_j if $|r(i,j)| > \delta$. As shown in lines 11–19, it then computes the AUC scores and FA scores for each candidate feature and ranks the features in order of FA scores. The algorithm selects the first r features in line 20. Compared with the existing method in [11], SAFE has two advantages: (i) instead of using all features, SAFE evaluates the correlation between the features and removes the redundant features; (ii) SAFE ranks the features by evaluating the feature importance and selects only the most relevant features.

Algorithm 1 Feature Selection Algorithm.

Input: The feature set $F = \{f_1, f_2, ..., f_d\}$, the threshold of Pearson δ, the number of selected features r;
Output: The selected features F'
// Calculate the correlation and remove related features.
1: $F' \leftarrow \emptyset$;
2: **for** $(i = 0; i < d; i++)$ **do**
3: **for** $(j = i + 1; j < d; j++)$ **do**
4: $r_{(i,j)} \leftarrow compute\_PCC(f_i, f_j)$;
5: **if** $(|r_{(i,j)}| > \delta)$ **then**
6: remove feature f_j;
7: **end if**
8: **end for**
9: **end for**
10: $F' \leftarrow$ Redundant features removed (F);
// Calculate AUC score and rank feature
11: **for** $(\forall f_i \in F')$ **do**
12: $AUC(f_i) \leftarrow calcualte\_AUC(f_i)$
13: **if** $(AUC(f_i) < 0.5)$ **then**
14: $FA(f_i) = 1 - AUC(f_i)$
15: **else**
16: $FA(f_i) = AUC(f_i)$
17: **end if**
18: **end for**
19: $F' \leftarrow$ sort (F') in order of $FA(f_i)$
20: $F' \leftarrow$ top r features of F'
21: return F'

3.3. Threshold tuning

To determine the optimal threshold value for each selected feature, we use a simple threshold tuning method. The threshold tuning method searches a range of thresholds between the minimum and the maximum of the feature and evaluates the F_1 score for each threshold. The best threshold can be determined as the threshold that can maximize the F_1 score. An example of threshold tuning is shown in Fig. 3.

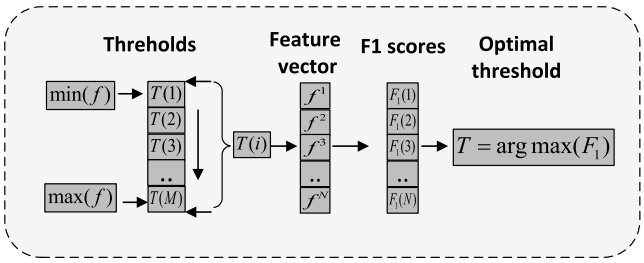


Fig. 3. An example of searching the optimal threshold for feature f.

Let M be the number of thresholds between the minimum and maximum values of feature f, and $T(i)$ be the ith threshold value, $i \in [1, M]$. Algorithm 2 shows the pseudocode of selecting the optimal threshold for feature f. Its computational complexity is $O(MN)$, where M is the number of thresholds and N is the number of flow samples. It predicts the label of the flow samples and computes the F_1 scores for all thresholds (lines 1–22). For a given threshold $T(i)$, if $AUC(f) \geqslant 0.5$, indicating that the values of the feature f in the benign flows are generally smaller than those in the attack flows, a flow sample is predicted as a benign flow if the feature's value does not exceed the threshold $T(i)$; otherwise, it is labeled as an attack flow (lines 4–11). However, if $AUC(f) < 0.5$, which means that the values of feature f in the benign flows are larger than those in the attack flows, we revise the label (lines 12–20). The algorithm calculates the F_1 score based on the true label and the predicted label for the threshold $T(i)$ (line 21). The flow sample prediction process stops after all M thresholds are iterated. The optimal threshold of feature f is chosen as the threshold with the highest F_1 score (lines 23–24).

3.4. Classification process

We stated that our feature selection can select the relevant features, but it still cannot identify which flow is truly an attack flow. In this part, our method is responsible for outputting the final prediction labels for the flows. A selected feature and the best threshold can be used as a classifier in the above processes; however, the classification accuracy and robustness can be further improved by adding more features with their corresponding optimal thresholds, i.e., increasing the number of classifiers. To do this, SAFE establishes an aggregated classification method based on the outputs of the selected features with the best thresholds and then weights the outputs to construct a linear aggregated classifier for classification (shown in Fig. 4).

Let $D(i, j)$ be the output predicted by the best threshold of the ith feature for the jth flow, where $D(i, j) = 1$ if the flow sample is predicted as an attack flow by this feature; otherwise, $D(i, j) = 0$. For r selected features, the output of the jth flow can be described as $L_j = \{(y_j, D(1, j), D(2, j), \ldots, D(r, j))\}$, where y_j is the true label. The linear aggregated classifier can be defined as:

$$L_j = \sum_{i}^{r} w_i D(i, j) + \varepsilon \tag{4}$$

where w_i is the weight of the ith output from the selected feature and ε is the bias. We use the sigmoid as the logistic function, which is defined as:

$$sigmoid(t) = \frac{1}{1 + e^{-t}} \tag{5}$$

Algorithm 2 The threshold tuning algorithm for feature f.

Input: The feature vector of the selected feature $f \in R^N$, AUC score of the feature $AUC(f)$, number of thresholds M, true label y;
Output: The optimal threshold α for feature f

$y_{pred} \leftarrow \emptyset$;
for $(i = 0; i < M; i++)$ **do**
  $T(i) \leftarrow min(f) + i \times (max(f) - min(f))$
  if $AUC(f) \geq 0.5$ **then**
    for $(j = 0; j < N; j++)$ **do**
      if $(f^j < T(i))$ **then**
        $y_{pred}[j] \leftarrow 0$;
      else
        $y_{pred}[j] \leftarrow 1$;
      end if
    end for
  else if $AUC(f) < 0.5$ **then**
    for $(j = 0; j < N; j++)$ **do**
      if $(f^j < T(i))$ **then**
        $y_{pred}[j] \leftarrow 1$;
      else
        $y_{pred}[j] \leftarrow 0$;
      end if
    end for
  end if
  $f1score(i) \leftarrow$ calculate f1 macro (y, y_{pred});
end for
$index = argmax(f1scores)$
$\alpha = M(index)$
return α

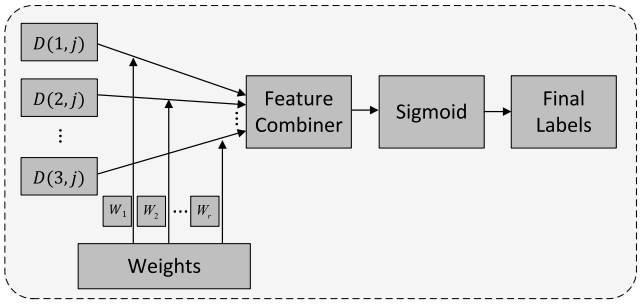


Fig. 4. The flowchart of the feature-based classification.

where $sigmoid(t) \in [0, 1]$, and the cutoff is set 0.5. Therefore, an attack flow label can be set if $sigmoid(t) > 0.5$. The final label of the flow is:

$$y'_j = \begin{cases} 1, & \text{if } sigmoid(t) > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Algorithm 3 describes the details of the feature binarisation process of the SAFE. Its computational complexity is $O(rN)$, where r is the number of selected features. The algorithm labels the flow samples and outputs the binary representation for all selected features. This is done by investigating the AUC scores and comparing the relationship between the feature values of a flow and the optimal thresholds. For example, for a selected feature f_i, if $AUC(f_i) > 0.5$ and the feature value $f[i][j]$ is smaller than the optimal threshold $\alpha[i]$, then the jth flow sample is labeled as a benign flow by the feature f_i. If $AUC(f_i) > 0.5$ and $f[i][j] > \alpha[i]$, then the flow f_j is predicted as an attack flow by this feature (lines 3–10). The labels are revised when $AUC(f_i) < 0.5$ (lines 11–19).

Algorithm 3 The feature binarisation process of the SAFE.

Input: The selected features $F' \in R^{r \times N}$, the optimal threshold of feature $\alpha \in R^r$, AUC of the features $AUC \in R^r$;
Output: The binary representation of the features $D \in R^{r \times N}$.

1: $D[][] \leftarrow \emptyset$;
2: **for** $(i = 0; i < r; i++)$ **do**
3: **if** $AUC(f_i) > 0.5$ **then**
4: **for** $(j = 0; j < N; j++)$ **do**
5: **if** $(f[i][j] < \alpha[i])$ **then**
6: $D[i][j] \leftarrow 0$;
7: **else**
8: $D[i][j] \leftarrow 1$;
9: **end if**
10: **end for**
11: **else if** $AUC(f_i) < 0.5$ **then**
12: **for** $(j = 0; j < N; j++)$ **do**
13: **if** $(f[i][j] < \alpha[i])$ **then**
14: $D[i][j] \leftarrow 1$;
15: **else**
16: $D[i][j] \leftarrow 0$;
17: **end if**
18: **end for**
19: **end if**
20: **end for**
21: return D

After the flows are labeled by all selected features, the algorithm learns the weights for the outputs of the selected features. Algorithm 4 shows the details of the training of the weights and the bias using logistic regression method. This algorithm uses the binary representation of the selected features D and the true label y to train the weights W and the bias ε. Its computational complexity is $O(rN)$.

Algorithm 4 The training logistic regression of the SAFE.

Input: The binary features $D \in R^{r \times N}$, true label $y \in R^N$;
Output: The weight $W \in R^r$ and the bias ε for Logistic Regression.

$W, \varepsilon \leftarrow$ applying logistic regression (D, y)
return W, ε

Algorithm 5 shows the details of the testing algorithm in the feature-based classification process of SAFE. Its computational complexity is $O(r)$. For a tested flow j, SAFE weights the outputs from the features and yields the final label using the sigmoid function. If the final label is 1, then the flow is identified as an attack flow; otherwise, it is a benign flow. Note that, the feature selection and binaristion of the testing flow are based on the training data through Algorithm 1 to 3.

Algorithm 5 The testing logistic regression of the SAFE.

Input: The weight $W \in R^r$ and the bias ε, the binary features $D \in R^{r \times N}$;
Output: y' (0: a benign flow; 1: an attack flow)

for $(j = 0; j < N; j++)$ **do**
  $L_j = \sum_i^r w[i]D[i][j] + \varepsilon$
  $y'_j \leftarrow sigmoid(L_j)$
end for

Table 3
The details of the attack and the benign flows in the three datasets.

Name	# of attackers	# of flows	# of benign	# of attack
SYN	106	221,334	48,591	172,743
LowRate	135	204,539	200,000	4539
Mirai	27,264	230,272	200,000	30,272

4. Experimental results

4.1. Datasets

We now perform several experiments using real datasets with a ground truth set of flows to validate our method. In this section, one IoT DDoS and two sophisticated DDoS attack datasets are used and the details are presented as follows.

- Mirai: The dataset is from IMPACT [32], and it contains DDoS attack traffic launched by a IoT botnet, Mirai [2]. There are 27,264 attackers, and the victim address is 130.152.184.2. The dataset contains 30,272 attack flows from October 1, 2016, to October 31, 2016.
- SYN flooding: The dataset is from the IMPACT, and it contains a SYN flooding attack and background traffic on November 5, 2009 [32]. In this dataset, 106 attackers (e.g., IP address 19.202.221.71) send SYN flooding to a victim (IP address 172.28.4.7). The dataset provides us with a set of 221,334 flows with 172,743 attack flows included.
- LowRate: The dataset is from the Center for Applied Internet Data Analysis (CAIDA) [33], and it contains a low rate DDoS attack on August 5, 2007 [13]. We observe that 135 attackers are aiming at the victim whose IP address is 71.126.222.64. The dataset contains 4539 attack flows.

In order to design an effective and efficient detection algorithm for real DDoS attacks, it is necessary to analyze the characteristics of features in both attack flows and benign flows. Therefore, in this paper, we merged LowRate and Mirai, which contain the attack flows, with WIDE, which contains the benign flows, to generate the datasets that can reflect the characteristics of features in both attack flows and benign flows. This merging strategy is commonly recognized and widely used in the research community [13,34], and [35]. The WIDE dataset is collected from the MAWI Working Group of the WIDE project. We use the dataset as part of "a Day in the Life of the Internet" (DITL) project [36], which measures the daily events on the Internet. A total of 200,000 flows are randomly chosen as background flows on December 2, 2015. In the following experiments, we refer to the three datasets as SYN, LowRate, and Mirai. Thus, we have three classification tasks, and the details are presented in Table 3.

4.2. Experimental settings

We use a 64-bit Windows 10 system with an Intel(R) Core(TM) i7-8650U CPU @1.90 GHz and 16 GB RAM to deploy our method. We implement the methods in Python 3.7.

4.3. Evaluation

For a fair comparison, we normalize feature set x into the same range [0, 1] using the min–max normalization:

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)} \tag{7}$$

where $max(x)$ and $min(x)$ are the corresponding maximum and minimum features, respectively.

Imbalanced data distribution is the characteristic of DDoS detection in practice. Due to the imbalance distribution between the attack flows and the benign flows, using an accuracy metric ($\frac{Number\ of\ Correct\ predictions}{Total\ Population}$) to evaluate the classification performance may cause bias to the majority class, e.g., benign flow in this experiment. For example, in the flow classification of the LowRate data, the imbalance ratio between the benign flows and the attack flows is 97.7% : 2.3%, which indicates that the default accuracy metric of the benign flows is 97.7%, although the method has not learned anything from the features. Therefore, we employ F_1 score, which is defined as the harmonic average of precision and recall; then, we weight the method on the performance across both majority and minority classes using macro F_1. The F_1 score is defined as:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{8}$$

where *precision* is the ratio of correctly identified attack flows to predicted attack flows, and *recall* is the ratio of identified attack flows to actual attack flows:

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

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

where the true positive (TP) is the attack flow that is detected as attack flow, false positive (FP) is the benign flow that is detected as attack flow, and false negative (FN) is the attack flow that is detected as benign flow. $F_1 \in [0, 1]$, where 1 is the best result and 0 is the worst result. Given k predicted classes, the macro F_1 score is the unweighted average over all classes as:

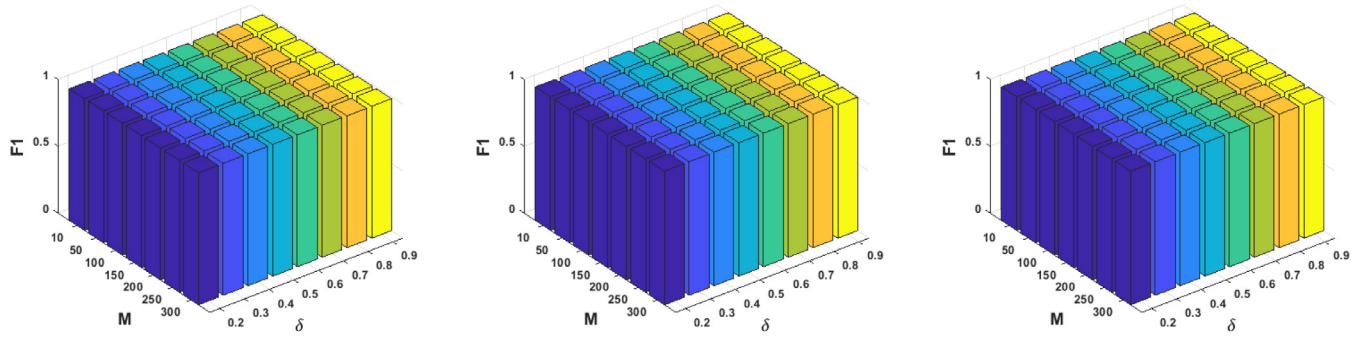
$$\bar{F_1} = \frac{1}{k} \sum_{i=1}^{k} \frac{2 \times Precision_i \times Recall_i}{Precision_i + Recall_i}. \tag{11}$$

4.4. Classification results

We conduct microbenchmark and comparison experiments to evaluate the performance of SAFE. Microbenchmark experiments study the effectiveness and performance of the proposed method with different parameters under various attacks, while comparison experiments demonstrate the effectiveness and efficiency of SAFE.

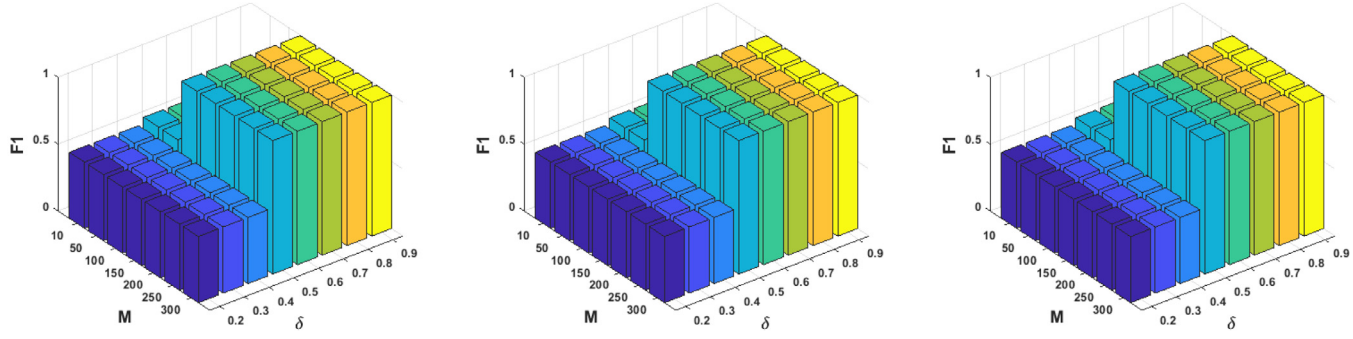
4.4.1. Effect of δ on accuracy

We evaluate the classification performance F_1 scores when the threshold of PCC δ varies. The threshold δ is an important parameter that determines the number of candidate features. A lower δ results in only a small number of features to be selected in the candidate feature set, while a higher δ leads to more candidate features. Figs. 5, 6, and 7 show the impact of δ on the classification performance of the three datasets. We observe that when δ increases, the F_1 scores increase. This is because when δ is small, only the most independent features are selected, and these features may not be informative. As δ increases, the number of candidate features increases, and thus, we can select the relevant features with higher *FA* scores. For example, in the LowRate flow classification with $r = 2$ and $M = 300$ (shown in Fig. 6b), when $\delta = 0.2$, five candidate features, F1, F5, F11, F15, and F21, pass the selection of PCC, and the *FA* scores are 0.9579, 0.9485, 0.9012, 0.8872, and 0.5725, respectively. The final F_1 score is only 0.4943. If we set $\delta = 0.7$, 18 features are left, and the top five features are F8, F1, F20, F5, F11, and F16 with much higher *FA* scores, which are 0.9944, 0.9579, 0.9502, 0.9485, and 0.9012, respectively. The final F_1 score significantly increases to 0.9980. However, a higher δ results in more redundant features selected. To balance the classification performance and redundancy of the features, we set $\delta = 0.7$. The details of the PCC values of the three datasets are shown in Fig. 8.



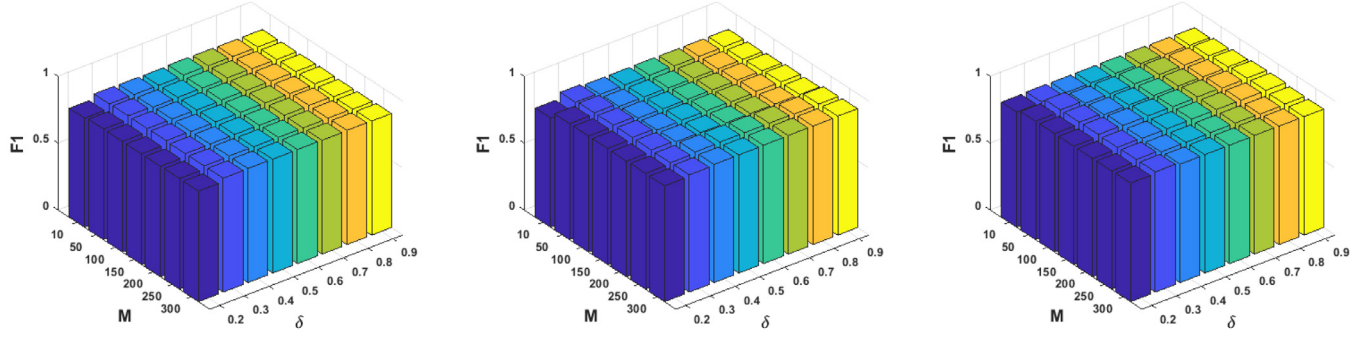
(a) $r = 1$ **(b)** $r = 2$ **(c)** $r = 3$

Fig. 5. The F_1 scores of SAFE in the SYN flooding attack flow classification.



(a) $r = 1$ **(b)** $r = 2$ **(c)** $r = 3$

Fig. 6. The F_1 scores of SAFE in the LowRate attack flow classification.



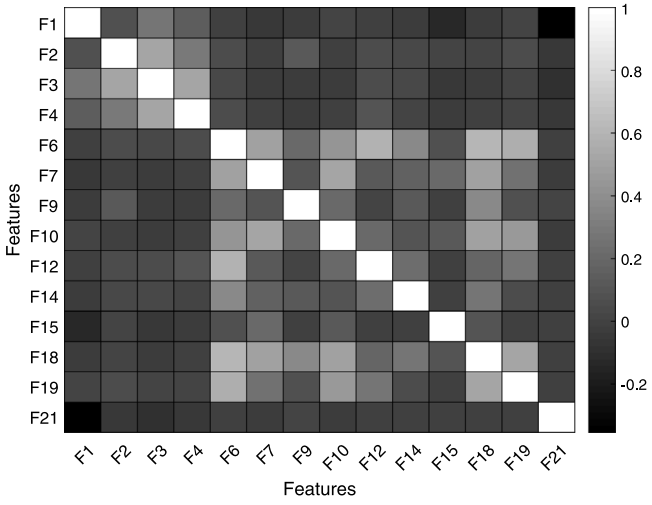
(a) $r = 1$ **(b)** $r = 2$ **(c)** $r = 3$

Fig. 7. The F_1 scores of SAFE in the Mirai attack flow classification.

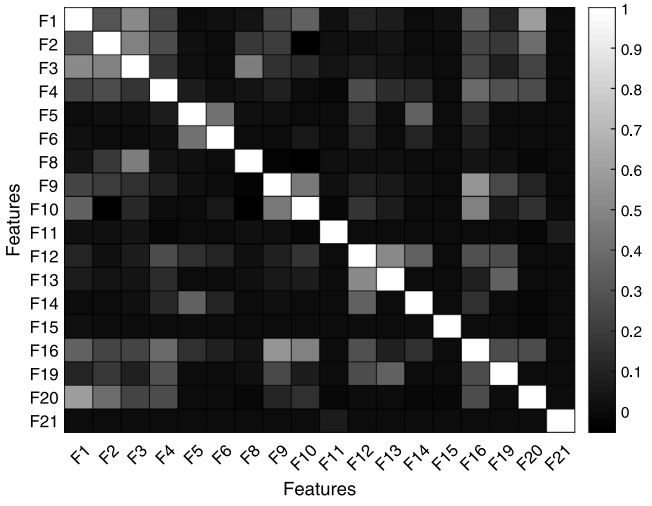
4.4.2. Effect of r on accuracy

We first present the details of the selected features and then evaluate the performance when the number of selected features varies. Fig. 9 shows the features ranked by the *FA* scores when $\delta = 0.7$ for the three attack flow classifications, which clearly shows that for the SYN flooding attack flow classification, the top three features are F1 (*src port*), F9 (fwd *pkts len std*), and F19 (fwd *max ACK int*), and the *FA* scores are 0.9986, 0.9925, and 0.9907. For LowRate, the top three features are F8 (fwd *pkts len min*), F1 (*src port*), and F20 (fwd *SYN num*), where the *FA* scores are 0.9944, 0.9579, and 0.9502. For Mirai, the top three features are F14 (fwd *psh num*), F20 (fwd *SYN num*), and F1 (*src port*), where the *FA* scores are 0.9506, 0.9495, and 0.9381. Note that the feature selection applies local information of the specific types of DDoS attack flows for computing an optimal feature subset. The selected feature subset varies over the attack types, as a consequence of the feature redundancy (i.e., δ) and feature importance (i.e., *FA* score). These features provide specialized information relevant to the specific DDoS attack strategies.

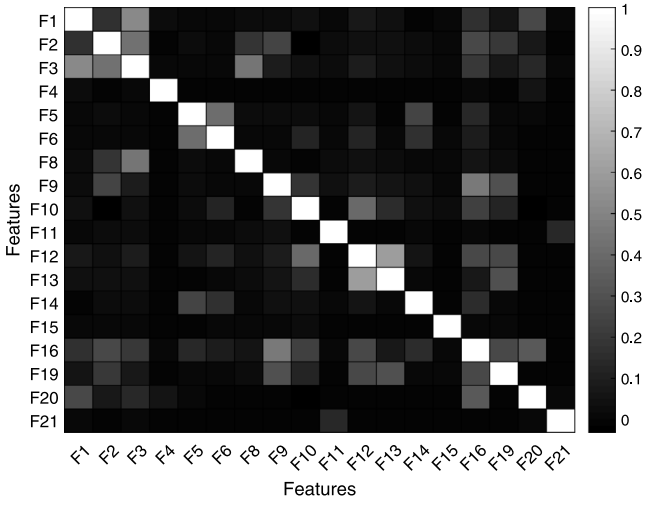
We evaluate the detection accuracy when the number of selected features varies. To select the most informative features and improve efficiency, we choose the number of features ranging from 1 to 3. We set δ to 0.7, and the number of thresholds is set to 200 and 300. As shown in Table 4, the F_1 scores of the three classifications increase with the number of selected features. Moreover, the performance is very high even when $r = 1$. This is because the selected features are ranked according to the *FA* scores, and the first feature is the most informative. The F_1 scores can achieve 0.98, 0.99, and 0.85 for the three classifications when only one feature is selected for both $M = 200$ and $M = 300$.



(a) 14 features are left in SYN flooding.



(b) 18 features are left in LowRate.



(c) 18 features are left in Mirai.

Fig. 8. The PCC values of the features when $\delta = 0.7$ for the three attacks.

Table 4
The F_1 scores for the three classifications when the number of features increase.

r	$M = 200$			$M = 300$		
	SYN	LowRate	Mirai	SYN	LowRate	Mirai
1	0.9852	0.9975	0.8521	0.9863	0.9974	0.8508
2	0.9978	0.9964	0.8651	0.9977	0.9980	0.8865
3	0.9979	0.9970	0.8949	0.9979	0.9969	0.8868

4.4.3. Effect of M on accuracy

We evaluate the detection performance on the number of thresholds M. Figs. 5, 6, and 7 show the performance of the three attack flow identifications when the number of thresholds varies. We observe that when the number of thresholds increases, the F_1 scores of all three datasets increase. This is reasonable, as we can select the optimal thresholds from a larger number of thresholds, which leads to more iterations to search for the highest F_1 score for each selected feature. However, the experimental results show that $M = 300$ is sufficient to find the optimal thresholds for all selected features in the three types of attack flow classification.

4.5. Experimental comparison

To validate the effectiveness and efficiency of our method, we conduct comparison experiments with two state-of-the-art methods: "Guard" [11] and MIR [10]. "Guard" identifies DDoS attack flows using an LSTM model with 40 features while MIR classifies the attack flows according to the transmission rate and the number of distinct messages. In addition, we also compare our method with three classification methods: K nearest neighbors (KNN) [37], deep learning (DP) [38], and support vector machine (SVM) [39].

We investigate the threshold of MIR. Due to the challenge of finding the optimal threshold, we search 300 thresholds ranging from the minimum to the maximum and report the highest F_1 as the best classification result, i.e., $M = 300$. For Guard, KNN, DP, and SVM, we randomly split 70% of the flows into training data and 30% into testing data. For KNN, DP, and SVM, we use the top three features selected by our feature selection method for each attack (shown in Fig. 9), i.e., $r = 3$ and $\delta = 0.7$. For a fair comparison, we use the default parameters for KNN (brute tree), DP, and SVM (rbf kernel) in sklearn [40]. We also set $\delta = 0.7$, $r = 3$ and $M = 300$ for SAFE based on the experimental results for the three attacks shown in Section 4.4 and report the average results over 10 trials.

4.5.1. Comparison on accuracy

To evaluate the performance globally, we test the compared methods when the number of flows (i.e., the percentage of the whole dataset) increases. Fig. 10 illustrates the classification performance of the compared methods for the three datasets with 95% confidence interval.

For MIR, SAFE outperforms the MIR method on the three datasets for all data size scenarios. For example, when we use 80% of the whole dataset for comparison, the F_1 scores of our method on the three datasets are 0.99, 0.98, and 0.99, significantly higher than the MIR method, which are 0.93, 0.58, and 0.37. The reason is that the *FA* score ranking of the features helps select the most relevant features. The average *FA* scores of the selected three features in our method are both 0.99 for all three datasets (shown in Fig. 9) while the *FA* scores of the MIR are only 0.99, 0.86, and 0.61 in the three datasets, which indicates that the selected features in SAFE are better than those in MIR. Therefore, our method outperforms MIR in terms of classification accuracy.

KNN, DP, SVM, and SAFE have similar performances in the SYN flooding attack flow classification, which are slightly better than

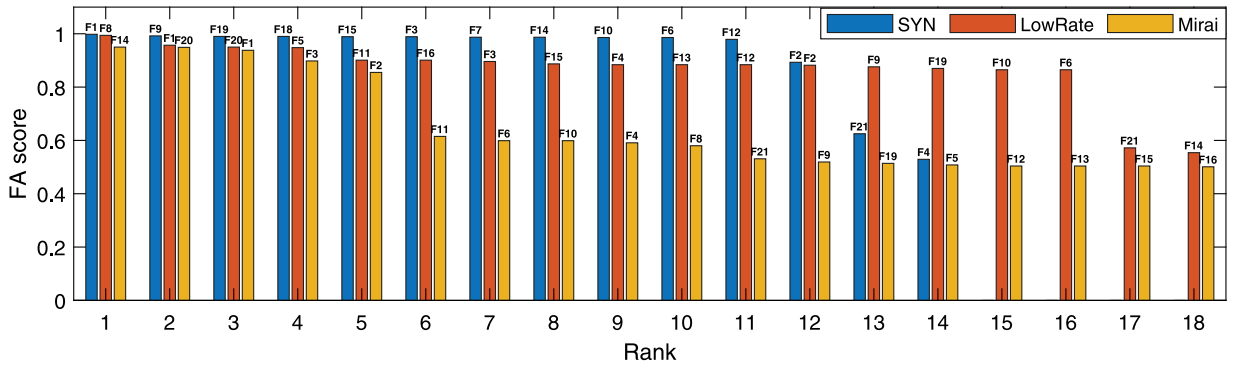
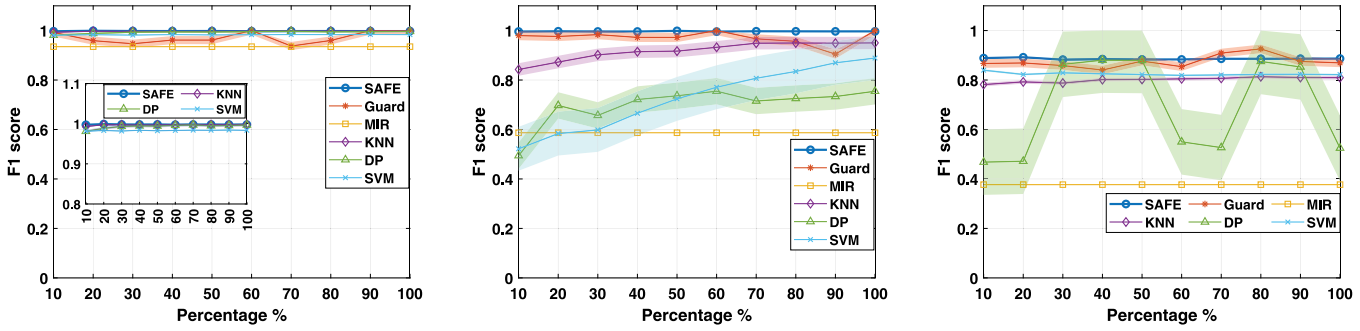
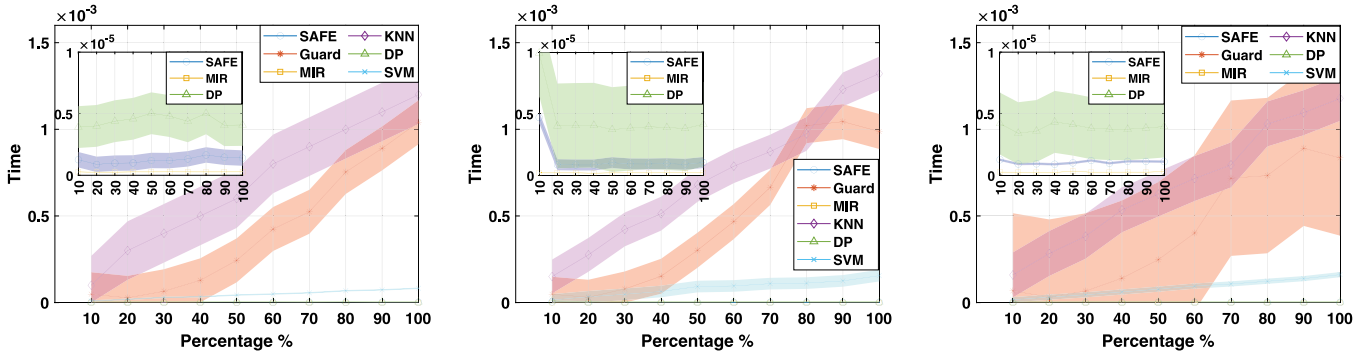


Fig. 9. The FA scores and the rank of the selected features when $\delta = 0.7$ for the three datasets.



(a) SYN **(b)** LowRate **(c)** Mirai

Fig. 10. Comparison accuracy on the three datasets. In (a), we provide a zoomed view for SAFE, KNN, DP, and SVM.



(a) SYN **(b)** LowRate **(c)** Mirai

Fig. 11. Comparison of testing time on the three datasets. In each figure, we provide a zoomed view for SAFE, MIR, and DP.

the Guard method (shown in Fig. 10a). However, one can see that our method is stable and generally higher than KNN, DP, SVM, and Guard in the LowRate and Mirai attack scenarios (shown in Figs. 10b and 10c). This is because these methods rely on large training flow samples and perform worse when the data size is small. For example, in the LowRate attack flow classification, when the flow sample percentage is 50%, the F_1 scores of Guard, KNN, DP, and SVM are 0.97, 0.91, 0.73, and 0.72, respectively, lower than SAFE, which is 0.98. On the Mirai dataset, the performance of the proposed method is close to that of Guard, which is better than those of KNN and SVM by approximately 8 percent. The DP performance fluctuates greatly, and the average F_1 score is 0.68. The results show that SAFE has better performance than the state-of-the-art methods in terms of classification accuracy even when the datasets are small.

4.5.2. Comparison on detection time

We evaluate the time consumption of the compared methods. Since the training phase is usually offline in practice, we compare the time cost for all methods in the testing phase. In the testing phase, the time consumption is from the processes of the best threshold matching and the weight matching for the selected features, i.e., Algorithms 3 and 4. The time complexity of these modules is linear to r, where in our experiment, $r = 3$ is the number of selected features.

Fig. 11 shows the testing time consumption of the six competing methods for each flow with 95% confidence interval on the three datasets in our experimental environment. We can see that except for MIR, SAFE significantly outperforms the other four methods. For example, in the LowRate dataset when the flow sample percentage is 50%, the time used for each flow in SAFE is less than that of the DP method by approximately 4 times and that of the Guard method by approximately 3×10^2 times. This

Table A.1
Full list of the features used in the experiments.

Features		
F1. Flow ID	F2. Src IP	F3. Src port
F4. Dst IP	F5. Dst port	F6. Protocol
F7. Flow Duration	F8. Tot Fwd Pkts	F9. Tot Bwd Pkts
F10. Tot Len Fwd Pkts	F11. Tot Len Bwd Pkts	F12. Fwd Pkt Len Max
F13. Fwd Pkt Len Min	F14. Fwd Pkt Len Mean	F15. Fwd Pkt Len Std
F16. Bwd Pkt Len Max	F17. Bwd Pkt Len Min	F18. Bwd Pkt Len Mean
F19. Bwd Pkt Len Std	F20. Flow Bytes per second	F21. Flow Pkts per second
F22. Flow IAT Mean	F23. Flow IAT Std	F24. Flow IAT Max
F25. Flow IAT Min	F26. Fwd IAT Tot	F27. Fwd IAT Mean
F28. Fwd IAT Std	F29. Fwd IAT Max	F30. Fwd IAT Min
F31. Bwd IAT Tot	F32. Bwd IAT Mean	F33. Bwd IAT Std
F34. Bwd IAT Max	F35. Bwd IAT Min	F36. Fwd PSH Flags
F37. Fwd Header Len	F38. Bwd Header Len	F39. Fwd Pkts per second
F40. Bwd Pkts per second	F41. Fwd entropy time IAT	F42. Bwd entropy time IAT
F43. Fwd entropy packet size	F44. Bwd entropy packet size	F45. Fwd DNS num
F46. Bwd DNS num	F47. Fwd DNS rate	F48. Bwd DNS rate
F49. Fwd max DNS IAT	F50. Bwd max DNS IAT	F51. Fwd min DNS IAT
F52. Bwd min DNS IAT	F53. Fwd var DNS IAT	F54. Bwd var DNS IAT
F55. Fwd TCP num	F56. Bwd TCP num	F57. Fwd TCP rate
F58. Bwd TCP rate	F59. Fwd max TCP IAT	F60. Bwd max TCP IAT
F61. Fwd min TCP IAT	F62. Bwd min TCP IAT	F63. Fwd var TCP IAT
F64. Bwd var TCP IAT	F65. Fwd ICMP num	F66. Bwd ICMP num
F67. Fwd ICMP rate	F68. Bwd ICMP rate	F69. Fwd max ICMP IAT
F70. Bwd max ICMP IAT	F71. Fwd min ICMP IAT	F72. Bwd min ICMP IAT
F73. Fwd var ICMP IAT	F74. Bwd var ICMP IAT	F75. Fwd HTTP num
F76. Bwd HTTP num	F77. Fwd HTTP rate	F78. Bwd HTTP rate
F79. Fwd max HTTP IAT	F80. Bwd max HTTP IAT	F81. Fwd min HTTP IAT
F82. Bwd min HTTP IAT	F83. Fwd var HTTP IAT	F84. Bwd var HTTP IAT
F85. Fwd ACK num	F86. Bwd ACK num	F87. Fwd ACK rate
F88. Bwd ACK rate	F89. Fwd max ACK IAT	F90. Bwd max ACK IAT
F91. Fwd min ACK IAT	F92. Bwd min ACK IAT	F93. Fwd var ACK IAT
F94. Bwd var ACK IAT	F95. Fwd SYN ACK num	F96. Bwd SYN ACK num
F97. Fwd SYN num	F98. Bwd SYN num	F99. Fwd SYN rate
F100. Bwd SYN rate	F101. Fwd max SYN IAT	F102. Bwd max SYN IAT
F103. Fwd min SYN IAT	F104. Bwd min SYN IAT	F105. Fwd var SYN IAT
F106. Bwd var SYN IAT	F107. Fwd ICMP unreachable num	F108. Bwd ICMP unreachable num
F109. Fwd ICMP unreachable rate	F110. Bwd ICMP unreachable rate	F111. Fwd max ICMP unreachable IAT
F112. Bwd max ICMP unreachable IAT	F113. Fwd min ICMP unreachable IAT	F114. Bwd min ICMP unreachable IAT
F115. Fwd var ICMP unreachable IAT	F116. Bwd var ICMP unreachable IAT	F117. Fwd RST num
F118. Bwd RST num	F119. Fwd ICMP time out num	F120. Bwd ICMP time out num

is because the testing computational complexity of the proposed method is $O(r)$ as aforementioned, while it is $O(rw)$ for the deep learning methods DP and Guard, where w is the number of weights. Therefore, SAFE is more efficient than the deep learning methods. Also, KNN and SVM perform much slower than our proposed method SAFE, which costs approximately 6×10^2 times and 9×10^1 more than SAFE, respectively. On the Mirai dataset, SAFE can still achieve a smaller time cost than the other four methods, in which it is 4 times faster than DP, 7×10^1 faster than SVM, 2×10^2 faster than Guard, and 6×10^2 faster than KNN when 50% percentage of the flows is used. When the number of flows increases, the time costs of SVM, Guard, and KNN increase nearly linearly. Similar results can be found in the SYN dataset. The experimental results show that MIR and our method have better classification performance than the four compared methods in terms of classification efficiency.

In summary, compared with MIR, the proposed method can achieve higher accuracy in attack flow classification, while it can also classify the attack flow more effectively and efficiently compared to Guard, KNN, DP, and SVM.

5. Conclusion

To defend against DDoS attacks, it is essential to have an effective system that can quickly distinguish attack flows from benign flows. In this paper, we propose a DDoS attack flow classification system named SAFE by selecting the optimal features. First, to select the most informative features, we remove the redundant features and select the most informative features by ranking the feature importance. Second, to find the best thresholds for each selected feature, we propose a threshold tuning method that can maximize the feature performance. Third, we combine the features and leverage an aggregated feature-based linear classifier to guarantee the effectiveness and robustness of SAFE. We carefully evaluate the parameters on one IoT DDoS and two sophisticated DDoS attacks. The comparison experiment results demonstrate that SAFE can more effectively and quickly identify attack flows than the compared methods.

CRediT authorship contribution statement

Lu Zhou: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Writing – original draft. **Ye Zhu:** Methodology, Validation, Formal analysis, Writing – review & editing. **Tianrui Zong:** Conceptualization, Investigation, Writing – review & editing. **Yong Xiang:** Conceptualization, Methodology, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix

See Table A.1

References

[1] Netscout's 14th annual worldwide infrastructure security report, 2021, https://www.netscout.com/report/ (accessed 5 September 2021).

[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. Halderman, L. Invernizzi, M. Kallitsis, et al., Understanding the mirai botnet, in: 26th {*USENIX*} Security Symposium ({*USENIX*} Security 17), 2017, pp. 1093–1110.

[3] K. Kalkan, L. Altay, G. Gür, F. Alagöz, JESS: Joint entropy-based DDoS defense scheme in SDN, IEEE J. Sel. Areas Commun. 36 (10) (2018) 2358–2372, http://dx.doi.org/10.1109/JSAC.2018.2869997.

[4] M. Ahmed, S. Ullah, H. Kim, Statistical application fingerprinting for DDoS attack mitigation, IEEE Trans. Inf. Forensics Secur. 14 (6) (2019) 1471–1484, http://dx.doi.org/10.1109/TIFS.2018.2879616.

[5] L. Zhu, X. Tang, M. Shen, X. Du, M. Guizani, Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks, IEEE J. Sel. Areas Commun. 36 (3) (2018) 628–643, http://dx.doi.org/10.1109/JSAC.2018.2815442.

[6] Z. Liu, Y. Cao, M. Zhu, W. Ge, Umbrella: Enabling ISPs to offer readily deployable and privacy-preserving ddos prevention services, IEEE Trans. Inf. Forensics Secur. 14 (4) (2019) 1098–1108, http://dx.doi.org/10.1109/TIFS.2018.2870828.

[7] J. Zheng, Q. Li, G. Gu, J. Cao, D. Yau, J. Wu, Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis, IEEE Trans. Inf. Forensics Secur. 13 (7) (2018) 1838–1853, http://dx.doi.org/10.1109/TIFS.2018.2805600.

[8] A. Wang, W. Chang, S. Chen, A. Mohaisen, Delving into internet DDoS attacks by botnets: Characterization and analysis, IEEE/ACM Trans. Netw. 26 (6) (2018) 2843–2855, http://dx.doi.org/10.1109/TNET.2018.2874896.

[9] N. Ravi, S. Shalinie, Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture, IEEE Internet Things J. 7 (4) (2020) 3559–3570, http://dx.doi.org/10.1109/JIOT.2020.2973176.

[10] V. Matta, M. Di Mauro, M. Longo, DDoS Attacks with randomized traffic innovation: Botnet identification challenges and strategies, IEEE Trans. Inf. Forensics Secur. 12 (8) (2017) 1844–1859, http://dx.doi.org/10.1109/TIFS.2017.2692685.

[11] Y. Jia, F. Zhong, A. Alrawais, B. Gong, X. Cheng, Flowguard: An intelligent edge defense mechanism against IoT DDoS attacks, IEEE Internet Things J. 7 (10) (2020) 9552–9562, http://dx.doi.org/10.1109/JIOT.2020.2993782.

[12] M. Bhuyan, D. Bhattacharyya, J. Kalita, Network anomaly detection: Methods, systems and tools, IEEE Commun. Surv. Tutor. 16 (1) (2014) 303–336, http://dx.doi.org/10.1109/SURV.2013.052213.00046.

[13] Y. Xiang, K. Li, W. Zhou, Low-rate DDoS attacks detection and traceback by using new information metrics, IEEE Trans. Inf. Forensics Secur. 6 (2) (2011) 426–437, http://dx.doi.org/10.1109/TIFS.2011.2107320.

[14] X. Yuan, C. Li, X. Li, Deepdefense: Identifying ddos attack via deep learning, in: 2017 IEEE Int. Conf. on Smart Comput. (SMARTCOMP), 2017, pp. 1–8, http://dx.doi.org/10.1109/SMARTCOMP.2017.7946998.

[15] R. Ujjan, Z. Pervez, K. Dahal, A. Bashir, R. Mumtaz, J. González, Towards sflow and adaptive polling sampling for deep learning based DDoS detection in SDN, Future Gener. Comput. Syst. 111 (2020) 763–779, http://dx.doi.org/10.1016/j.future.2019.10.015.

[16] S. Rehman, M. Khaliq, S. Imtiaz, A. Rasool, M. Shafiq, A. Javed, Z. Jalil, A. Bashir, DIDDOS: An approach for detection and identification of distributed denial of service (DDoS) cyberattacks using gated recurrent units (GRU), Future Gener. Comput. Syst. 118 (2021) 453–466, http://dx.doi.org/10.1016/j.future.2021.01.022.

[17] J. Cui, M. Wang, Y. Luo, H. Zhong, DDoS Detection and defense mechanism based on cognitive-inspired computing in SDN, Future Gener. Comput. Syst. 97 (2019) 275–283, http://dx.doi.org/10.1016/j.future.2019.02.037.

[18] Y. Xu, H. Sun, F. Xiang, Z. Sun, Efficient DDoS detection based on K-FKNN in software defined networks, IEEE Access 7 (2019) 160536–160545, http://dx.doi.org/10.1109/ACCESS.2019.2950945.

[19] M. Tayyab, B. Belaton, M. Anbar, ICMPV6-based DoS and DDoS attacks detection using machine learning techniques, open challenges, and blockchain applicability: A review, IEEE Access 8 (2020) 170529–170547, http://dx.doi.org/10.1109/ACCESS.2020.3022963.

[20] G. Lucky, F. Jjunju, A. Marshall, A lightweight decision-tree algorithm for detecting DDoS flooding attacks, in: 2020 IEEE 20th Int. Conf. Software Qual. Reliab. and Security Companion (QRS-C), 2020, pp. 382–389, http://dx.doi.org/10.1109/QRS-C51114.2020.00072.

[21] S. Das, D. Venugopal, S. Shiva, F.T. Sheldon, Empirical evaluation of the ensemble framework for feature selection in DDoS attack, in: 2020 7th IEEE Int. Conf. Cyber Security and Cloud Comput. (CSCloud)/2020 6th IEEE Int. Conf. Edge Comput. and Scalable Cloud (EdgeCom), 2020, pp. 56–61, http://dx.doi.org/10.1109/CSCloud-EdgeCom49738.2020.00019.

[22] W.L. Costa, M.M. Silveira, D.A. Thelmo, R.L. Gomes, Improving DDoS detection in IoT networks through analysis of network traffic characteristics, in: 2020 IEEE Latin-American Conf. Commun. (LATINCOM), 2020, pp. 1–6, http://dx.doi.org/10.1109/LATINCOM50620.2020.9282265.

[23] D. Kshirsagar, S. Kumar, A feature reduction based reflected and exploited DDoS attacks detection system, J. Ambient Intell. Hum. Comput. http://dx.doi.org/10.1007/s12652-021-02907-5.

[24] A. Callado, C. Kamienski, G. Szabo, B. Gero, J. Kelner, S. Fernandes, D. Sadok, A survey on internet traffic identification, IEEE Commun. Surv. Tutor. 11 (3) (2009) 37–52, http://dx.doi.org/10.1109/SURV.2009.090304.

[25] L. Friedman, O.V. Komogortsev, Assessment of the effectiveness of seven biometric feature normalization techniques, IEEE Trans. Inf. Forensics Secur. 14 (10) (2019) 2528–2536, http://dx.doi.org/10.1109/TIFS.2019.2904844.

[26] Y. Xie, S. Yu, Monitoring the application-layer DDoS attacks for popular websites, IEEE/ACM Trans. Netw. 17 (1) (2009) 15–25, http://dx.doi.org/10.1109/TNET.2008.925628.

[27] P. Kumar, M. Tripathi, A. Nehra, M. Conti, C. Lal, SAFETY: Early detection and mitigation of TCP SYN flood utilizing entropy in SDN, IEEE Trans. Netw. Serv. Manage. 15 (4) (2018) 1545–1559, http://dx.doi.org/10.1109/TNSM.2018.2861741.

[28] Canadian institute for cybersecurity flow meter, 2017, https://www.unb.ca/cic/research/applications.html (accessed 20 December 2021).

[29] A. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (1997) 1145–1159, http://dx.doi.org/10.1016/S0031-3203(96)00142-2.

[30] X. Chen, M. Wasikowski, FAST: A roc-based feature selection metric for small samples and imbalanced data classification problems, in: 14th ACM SIGKDD Int. Conf. Knowl. Discovery and Data Min, 2008, pp. 124–132, http://dx.doi.org/10.1145/1401890.1401910.

[31] Z. Yang, B. Zhou, Y. Lei, Y. Ying, Stochastic hard thresholding algorithms for AUC maximization, in: 2020 IEEE Int. Conf. Data Min. (ICDM), 2020, pp. 741–750, http://dx.doi.org/10.1109/ICDM50108.2020.00083.

[32] Information marketplace for policy and analysis of cyber-risk & trust, 2021, https://www.impactcybertrust.org/dataset_view?idDataset=742 (accessed 8 September 2021).

[33] CAIDA DDoS attack dataset, 2007, https://www.caida.org/data/passive/ddos-20070804_dataset.xml (accessed 5 September 2021).

[34] M. Eskandari, Z.H. Janjua, M. Vecchio, F. Antonelli, Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices, IEEE Internet Things J. 7 (8) (2020) 6882–6897, http://dx.doi.org/10.1109/JIOT.2020.2970501.

[35] W. Zhou, W. Jia, S. Wen, Y. Xiang, W. Zhou, Detection and defense of application-layer DDoS attacks in backbone web traffic, Future Gener. Comput. Syst. 38 (2014) 36–46, http://dx.doi.org/10.1016/j.future.2013.08.002.

[36] A day in the life of the internet (DITL), 2019, http://www.caida.org/projects/ditl/ (accessed 5 September 2021).

[37] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inform. Theory 13 (1) (1967) 21–27, http://dx.doi.org/10.1109/TIT.1967.1053964.

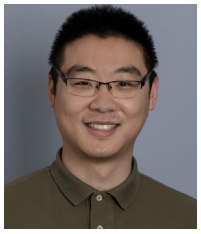
[38] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444, http://dx.doi.org/10.1038/nature14539.

[39] J. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (3) (1999) 293–300, http://dx.doi.org/10.1023/A:1018628609742.

[40] Scikit-learn, 2019, https://scikit-learn.org/stable/ (accessed 10 September 2021).

Lu Zhou received the M.S. degree from Wuhan Polytechnic University in 2014. He is currently a Ph.D. candidate with School of Information Technology, Deakin University, Australia. His research interests include machine intelligence, network security, especially in DDoS attacks detection.

Ye Zhu (M'17) received his Ph.D. degree in Artificial Intelligence with a Mollie Holman Medal for the best doctoral thesis of the year from Monash University (Australia) in 2017. He is a lecturer with the School of Information Technology, Deakin University, Australia since 2019. He joined Deakin University in 2017 as a research fellow of complex system data analytics. His research works focus on clustering analysis, anomaly detection, and their applications for pattern recognition and information retrieval.



Tianrui Zong received the B.Eng. degree in automation science and electrical engineering from BeiHang University, China in 2009, and the M.Sc. degree in signal processing and communications from the University of Edinburgh, the United Kingdom in 2010 and the Ph.D. degree in signal processing and communications from Deakin University, Australia in 2015. He is currently a Researcher with ZHONGTUKEXIN CO., LTD., China. His research interests include multimedia forensics and security, signal processing, and machine learning.

Yong Xiang (SM'12) received the Ph.D. degree in the Electrical and Electronic Engineering from The University of Melbourne, Australia. He is a Professor at the School of Information Technology, Deakin University, Australia. His research interests include information security and privacy, signal and image processing, data analytics and machine intelligence, Internet of Things, and blockchain. He has published 5 monographs, over 165 refereed journal articles, and numerous conference papers in these areas. He is the Senior Area Editor of IEEE Signal Processing Letters and the Associate Editor of IEEE Communications Surveys and Tutorials. He has served as Honorary Chair, General Chair, Program Chair, TPC Chair, Symposium Chair, and Track Chair for a number of international conferences.