# Toward Explainable and Adaptable Detection and Classification of Distributed Denial-of-Service Attacks

Yebo Feng(✉) and Jun Li

University of Oregon, Eugene, OR 97403, USA
{yebof,lijun}@cs.uoregon.edu

**Abstract.** By attacking (e.g., flooding) the bandwidth or resources of a victim (e.g., a web server) on the Internet from multiple compromised systems (e.g., a botnet), distributed Denial-of-Service (DDoS) attacks disrupt the services of the victim and make it unavailable to its legitimate users. Albeit studied many years already, the detection of DDoS attacks remains a troubling problem. In this paper, we propose a new, learning-based DDoS detection and classification method that is both explainable and adaptable. This method first utilizes a modified k-nearest neighbors (KNN) algorithm to detect DDoS attacks and then uses risk degree sorting with grids to classify traffic at a fine granularity. It uses a k-dimensional tree to partition the searching space that significantly improves its efficiency and shortens KNN query times. Moreover, compared with the previous DDoS detection and classification approaches, along with the detection results this method further generates risk profiles that provides users with interpretability for filtering DDoS traffic. Additionally, this method does not need to retrain the detection model in order to make it fit in a new network environment. Users can leverage a variety of prior knowledge to evolve the model. We evaluated this approach in both simulated environments and the real world, which shows that our approach is both effective and efficient. It achieves a 98.4% accuracy in detecting DDoS attacks with a delay of around 5 s.

**Keywords:** Distributed Denial-of-Service (DDoS) · DDoS detection · Anomaly detection · Machine learning · K-nearest neighbors (KNN)

## 1 Introduction

Distributed denial-of-service attacks (DDoS attacks) pose a severe security problem on today's Internet and can make specific servers, network infrastructures, or applications unavailable to their users. They typically operate by overwhelming the targeted machine or network resource with excessive requests, therefore preventing legitimate requests from being fulfilled [4]. Cisco released a white paper in March 2020 and indicated that the frequency of DDoS attacks had increased more than 2.5 times over the last three years. Moreover, the average

G. Wang et al. (Eds.): MLHat 2020, CCIS 1271, pp. 105–121, 2020.
https://doi.org/10.1007/978-3-030-59621-7_6

size of DDoS attacks is increasing steadily and approaching 1 Gbps, enough to paralyze most websites thoroughly [7].

Key to effectively preventing and mitigating DDoS attacks is prompt and accurate DDoS detection. Decades of research and industry efforts have led to a myriad of DDoS detection and classification approaches. In the beginning, rule-based and statistical DDoS detection approaches dominated this field. Such methods can hardly deal with sophisticated attacks, however. As machine learning algorithms evolve and mature, many researchers begin to harness such techniques on big data in detecting and classifying DDoS attacks. For instance, Suresh et al. [26] evaluated a variety of machine learning algorithms in detecting DDoS, including SVM, Naive Bayes, K-means, etc.; Yuan et al. [29] trained a recurrent deep neural network to discover DDoS activities. The results of such methods demonstrate their strong ability in extracting useful knowledge from massive training data to identify DDoS attacks.

However, the negative aspects of learning-based approaches are also apparent. Firstly, most of the learning-based approaches are inexplicable when making predictions. This black-box feature is troublesome because the unexplainable results are difficult for network administrators to review and verify, which could cause potential collateral damage when filtering DDoS traffic. Moreover, learning-based methods are not easily adaptable. The performance of such methods highly depends on the coverage and applicability of the training data, whereas DDoS attacks are diverse and highly dependent on the network environment. A confirmed DDoS attack in one environment may be considered as legitimate in another. Hence, it is difficult for almost all the learning-based approaches to convert a trained DDoS detection model to fit a new network environment.

To address these missing gaps, we design an explainable and adaptable DDoS traffic detection and classification method based on machine learning. It inputs flow-level network traffic and identifies DDoS attackers in two phases: *detection phase* and *classification phase*. In the detection phase, it employs the modified k-nearest neighbors (KNN) algorithm and a k-dimensional tree (KD tree) to detect DDoS attacks with the overall traffic profile. Here, the KD tree can significantly improve detection efficiency and accelerate the query process of KNN. Moreover, we convert the searching space of the KNN model into a semi-decision tree that can reduce the time complexity of traffic monitoring to $O(d)$ in most cases ($d$ is the depth of the semi-decision tree). Once a DDoS threat is detected, our approach enters the classification phase to classify traffic. It will sort the traffic sources based on risk degrees to reduce the collateral damage, then iteratively identify the malicious IP addresses until the traffic profile returns to a benign position in the KNN searching space.

Our approach offers interpretability and adaptability. During a DDoS attack, our approach not only outputs an alert message but also exports a *risk profile* to explain and quantify the attack. The risk profile is the shortest geometrical distance from the current traffic profile to a benign area in the KNN searching space, which provides the network administrators with an explainable summary about the current attack. Besides, users do not need to retrain the detection model to fit it with a new network environment. Our approach allows direct modifications on the KNN searching space and enables users to leverage a variety of prior knowledge to evolve the detection model.

We evaluated our approach in both simulated environments and the real world. We first trained and evaluated our detection model with representative DDoS datasets from public repositories in our simulation environment. The results indicate that the detection model can achieve an accuracy of 0.984 and a recall score of 0.985 when identifying popular DDoS attacks and classifying DDoS traffic, where the model also comes with explainable information to indicate the intensity and category of the attack. Furthermore, as this model is easily adaptable to a new environment, we then transferred the model (with merely some measurement data as input) to a real-world environment that consists of a 50 Gbps link in an ISP-level network. We successfully detected five real-world DDoS attacks from April to May 2020, which we verified with the ISP. The latency of our detection model is also low, even with huge amounts of throughput; with an throughput of 50 Gbps, for example, our approach can complete the detection in around five seconds.

The rest of this paper is organized as follows. After introducing the related works in Sect. 2, we describe the method design in Sect. 3, evaluate our approach in Sect. 4, and conclude this paper in Sect. 6. Of a particular note here is that this paper is an extended version of work published in [14].

## 2 Related Work

In this section, we present some representative DDoS attack detection and classification methods. According to their basic detection principles, we classify these existing methods as statistical approaches, rule-based approaches, and learning-based approaches.

### 2.1 Statistical Approaches

Statistical approaches detect DDoS attacks by exploiting statistical properties of benign or malicious network traffic. Generally, these approaches build a statistical model of normal or malicious traffic and then apply a statistical inference test to determine if a new instance follows the model [10]. For example, D-WARD [22] uses a predefined statistical model for legitimate traffic to detect anomalies in the bidirectional traffic statistics for each destination with periodic deviation analysis. Chen [11] proposed a DDoS detection method based on the two-sample t-test, which indicates that the SYN arrival rate of legitimate traffic follows the normal distribution and identifies a DDoS attack by testing the distribution compliance. Zhang et al. [31] proposed a detection method by applying the Auto Regressive Integrated Moving Average model on the available service rate of a protected server. A major drawback of statistical approaches is that as DDoS attacks evolve, DDoS traffic does not always show statistical significance in various aspects. Thus, statistical DDoS detection approaches may be inadequate for identifying modern DDoS attacks.

### 2.2 Rule-Based Approaches

Rule-based approaches formulate noticeable characteristics of known DDoS attacks and detect actual occurrences of such attacks based on those formulated characteristics. NetBouncer [27] detects illegitimate clients by conducting a set of legitimacy tests of the clients; If a client fails to pass these tests, it will be considered as malicious traffic sources until a particular legitimacy window expires. Wang et al. [28] detects DDoS with an augmented attack tree (AAT), which captures incidents triggered by DDoS traffic and the corresponding state transitions from the view of network traffic transmissions. Limwiwatkul et al. [20] detects ICMP, TCP and UDP flooding attacks by analyzing the packet headers with well-defined rules and conditions. However, it is difficult for all the rule-based approaches to detect unseen attacks. People have to summarize and formulate the features of all the possible DDoS attacks that could happen in the network environment before using such methods, which is hard to achieve in real scenarios.

### 2.3 Learning-Based Approaches

Over the past few years, more and more researchers began to leverage machine learning to model and detect DDoS attacks [8,13,15,17,18,21,25,30]. Some of these methods utilize unsupervised learning algorithms [9,19,32]. They do not require training before the detection but are sensitive to the selected features and the background traffic. On the other hand, the majority of the methods using supervised learning algorithms cannot provide the users with explainable detection results. Since the dominant machine learning algorithms such as linear regression, multilayer perceptron, and convolutional neural network are similar to black boxes, network administrators need to turn to the raw traffic to dig out information for reviewing the detection results before mitigating the attacks. Thus, although learning-based approaches are usually accurate in detecting DDoS attacks, they are not very reliable in real deployments. Besides, the applicability of these machine learning algorithms highly depends on the training data and environment. It is difficult to quickly transfer a detection model trained in one network environment to another network environment. Different from these previous learning-based approaches, Our approach focuses on the explainability and adaptability of the detection model.

## 3 Design

Our approach offers DDoS detection and classification at the victim end on a router that sees all the traffic toward and from the victim. Figure 1 illustrates its operational model. It inputs flow-level traffic data from the router that runs widely used traffic capture engines, such as NetFlow [12] and sFlow [24], and monitors the traffic for DDoS detection and classification.

Our approach works in two phases: detection phase and classification phase. It first detects whether there is a DDoS attack or not in the network during

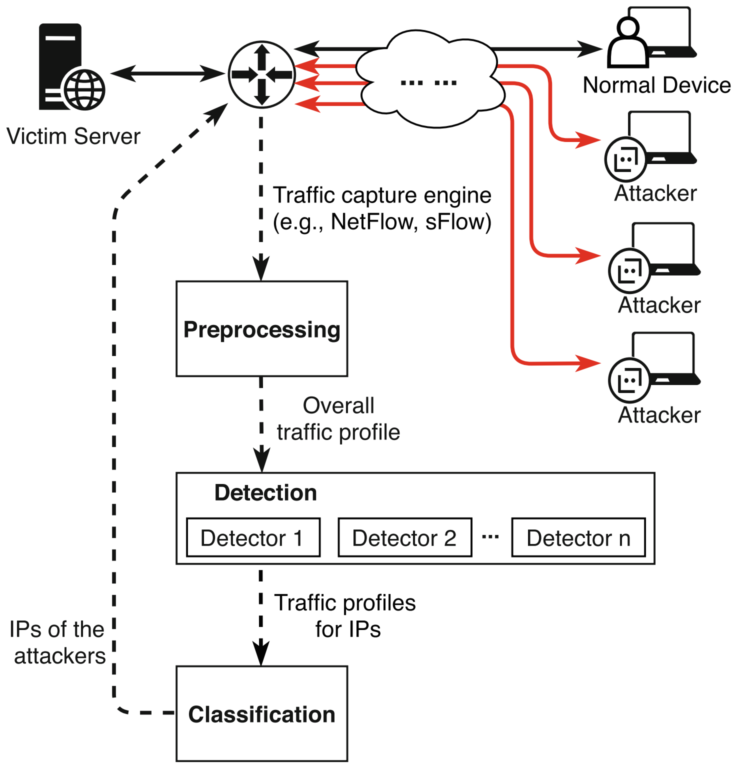

**Fig. 1.** Operational model of DDoS detection and classification.

the detection phase. To provide a comprehensive protection to the victim, our approach can employ multiple detectors, with each focusing on certain types of DDoS attacks. Once a DDoS attack is detected, it then enters the classification phase. It generates a traffic profile $\boldsymbol{p}$ for every individual IP address, classifies traffic at a fine granularity according to IP traffic profiles, and outputs malicious IP addresses for further actions.

Our approach monitors the traffic in batches. Each batch is a uniform time bin, $t$, which is also the most basic detection unit. In our implementation, we set each batch as 5 s. During each batch $t$, the preprocessing module of our approach extracts from the input data features to form different types of overall traffic profiles. A profile can be denoted as $\boldsymbol{S}$, with $\boldsymbol{S} = \{f_1, f_2, f_3, ..., f_n\}$, where $f_n$ denotes the value of the $n$-th feature during batch $t$. The features in $\boldsymbol{S}$ depend on the detectors we use, as each detector may need a different traffic profile with different features.

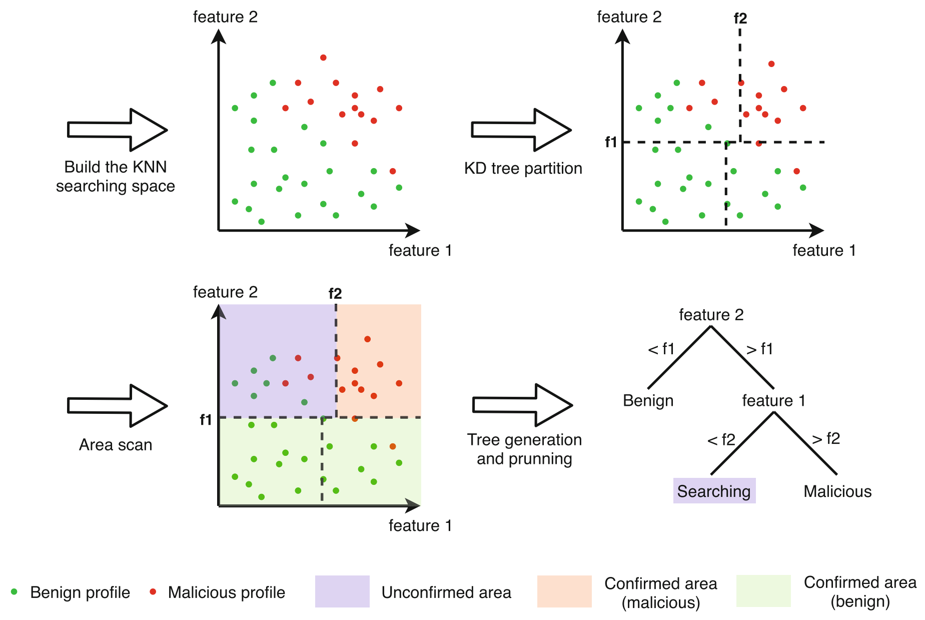

**Fig. 2.** A DDoS detector with the modified KNN algorithm and KD tree.

## 3.1 Detection Phase

The goal of the detection phase is to determine whether a DDoS attack is present or not according to the current traffic profile $\boldsymbol{S}$. We use the KNN algorithm to achieve this. The KNN algorithm is a non-parametric method used for classification, which finds the $k$ nearest neighbors of the traffic profile $\boldsymbol{S}$ and use their classifications to vote for the label of $\boldsymbol{S}$. This algorithm is simple, straightforward, and reliable. Users can also choose to build multiple KNN detection models to detect a variety of DDoS attacks, as Fig. 1 shows.

In our implementation, we constructed five detection models. For example, we chose six features to construct the traffic profile for the TCP SYN flood detection model, which are the number of TCP bytes, the number of TCP packets, the ratio of inbound TCP packet number to outbound TCP packet number, the number of SYN flags, the number of flows, and the number of PSH flags.

The reason for building multiple KNN models to tackle different attacks instead of building one complicated KNN model is to avoid the curse of dimensionality [16]. A detection model that can tackle different types of attacks usually needs to process data in high-dimensional spaces. However, the increase in the number of dimensions of a dataset can make the searching space sparser. We thus need much more training data to cover the searching space; otherwise the accuracy of the detection model would be unsatisfactory. As a solution to this problem, we construct multiple KNN models to cover different attacks, with each having only a few features.

The KNN algorithm has one weakness, however. Although it takes little time to train the model, the prediction requires a time complexity of $O(nlogn)$ to complete because it needs to enumerate the data points in the searching space to find $k$ nearest neighbors. Hence, we leverage the KD tree to partition the searching space, thus reducing the number of data points to enumerate. With the KD tree, whenever there is an incoming profile, we only need to search a sub-area to predict the result. Figure 2 shows a simple example where only two features are included in the training and prediction.

Furthermore, according to our experimental results, compared with legitimate traffic profiles, most DDoS profiles have relatively big differences. This leads to an interesting fact that most of the searching areas partitioned by the KD tree only have either benign traffic profiles or malicious traffic profiles. As shown by the red and green areas in Fig. 2, we define the searching area as a confirmed area if one type of the traffic profiles dominate the area and the number of any other type of traffic profiles is smaller than $k/2$. If the current traffic profile $S$ falls within a confirmed area, we can directly label the profile $S$ with the identity of the confirmed area without conducting any KNN queries. Thus, we convert the original KNN query process into a tree-like data structure. The detection module will only trigger the search of nearest neighbors when the traffic profile $S$ falls within an unconfirmed area. If DDoS attacks do not happen frequently, this tree-like data structure can reduce the time complexity for traffic monitoring to $O(d)$, where $d$ is the depth of the tree.

**Explainability:** Once an anomaly is detected, our approach not only outputs an alert message but also exports a value to explain and quantify the anomaly. Such value is the risk profile $\Delta$ ($\Delta = (m, \delta)$), which provides the network administrators with an explainable summary about the current attack. Here, $m$ is the name of the feature in the traffic profile $S$ that causes the anomaly. This attribute helps the network administrator determine the attack status and pinpoint the root cause. $\delta$ is the value by which feature $f_m$ needs to be reduced to make the traffic profile $S$ return back to a benign space. In other words, $\delta$ is the shortest distance from the current traffic profile to a legitimate traffic profile in the KNN searching space.

To figure out $m$ and calculate $\delta$, we need to calculate the average traffic profile $\overline{\boldsymbol{p}}$ for each distinct IP address first, where $\overline{\boldsymbol{p}} = \boldsymbol{S}/number\_of\_ips$. Then, we use Eq. 1 to calculate $\delta$.

$$\begin{aligned} &\boldsymbol{S_\Delta} = \boldsymbol{S} - \boldsymbol{L} \\ &\quad \delta = max(\frac{f_1^{(\boldsymbol{S_\Delta})}}{f_1^{(\overline{\boldsymbol{p}})}}, \frac{f_2^{(\boldsymbol{S_\Delta})}}{f_2^{(\overline{\boldsymbol{p}})}}, \frac{f_3^{(\boldsymbol{S_\Delta})}}{f_3^{(\overline{\boldsymbol{p}})}}, ..., \frac{f_n^{(\boldsymbol{S_\Delta})}}{f_n^{(\overline{\boldsymbol{p}})}}) \end{aligned} \tag{1}$$

Here, $\boldsymbol{L}$ is the closest traffic profile in a benign space, which can be found with the breadth-first search on the semi-decision tree; $f_n^{(\boldsymbol{S})}$ denotes the $n$-th feature in profile $\boldsymbol{S}$.

In a few cases, there can be more than one shortest distance, which means $\Delta = \{(m_1, \delta_1), (m_2, \delta_2), ..., (m_n, \delta_n)\}$. We consider the anomalies are caused by legitimate flash crowds under these circumstances, since the overall traffic is still in a reasonable shape. Hence, there could be a large surge of legitimate traffic focusing on specific hosts in the network and the detection program will turn over decisions to the network administrator for next measures.

## 3.2 Phase Two: Classification

The objective of the classification phase is to recognize the malicious IP addresses and output them for DDoS traffic filtering. Of a particular note is that the classification module will only be activated after some anomalies have been detected in the detection phase.

The design philosophy of the traffic classification is that the traffic profile $\boldsymbol{S}$ is currently in a malicious position, and we need to restrict the traffic from the most suspicious IP addresses so that the traffic profile can return to a benign area.

We conduct the classification for malicious sources by building a traffic profile $\boldsymbol{p}$ for each IP address. The profile $\boldsymbol{p}$ should have the same attributes as the overall traffic profile $\boldsymbol{S}$. The only difference is that the values of features in $\boldsymbol{p}$ are calculated from the traffic of each individual IP, while the values of features in $\boldsymbol{S}$ are calculated from the overall traffic in the network. Afterwards, we sort the IP addresses in the decreasing order of the risk degree, where the risk degree is a number indicates how suspicious an IP is. According to the risk profile $\varDelta$ ($\varDelta = (m, \delta)$) we obtained from the DDoS detection phase, we define the risk degree of an IP address as $f_m^{(\boldsymbol{p})}$. Finally, we conduct traffic filtering on IP addresses in such an order until the overall traffic profile returns to a benign area.

However, because legitimate clients sometimes may have significant risk degrees as well. Classifying the IP addresses only according to the risk degree may cause significant collateral damage. To address this issue, we also need to minimize the impact on other features of the overall traffic profile $\boldsymbol{S}$ when determining the malicious traffic sources. We consider this as an optimization problem with two constraints, which can be demonstrated as Eq. 2. Here, $G$ denotes the complete set of IP addresses we have seen during the DDoS attack, $G_m$ denotes the set of malicious IP addresses that the classification program will output, and $p^{(i)}$ denotes the traffic profile of the $i$th-IP.

$$\begin{aligned}\underset{G_m}{\operatorname{argmax}}\, f(G, G_m) &= \sum_{g \in G, g \notin G_m} \sum_{i \in g} \left\| \boldsymbol{p}^{(i)} \right\|_2 \\ &= \sum_{g \in G, g \notin G_m} \sum_{i \in g} \sqrt{\sum_{k=1}^{n} \left| f_k^{\boldsymbol{p}^{(i)}} \right|^2},\end{aligned} \tag{2}$$

$$\begin{aligned}\text{subject to: } &\sum_{g \in G_m} \sum_{i \in g} p_m^i \geq \delta, \\ &G_m \subseteq G.\end{aligned} \tag{3}$$

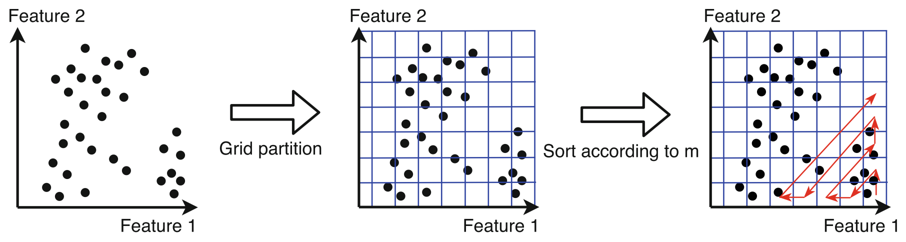

**Fig. 3.** An example of the classification process, where the dots in the coordinate system are traffic profiles for IP addresses and $m$ is Feature 1.

Equation 3 shows two constraints: (1) after eliminating all the traffic from malicious IP addresses, set $G_m$, the overall traffic profile should return to a benign area; (2) the malicious IP set $G_m$ should be the subset of the complete IP set $G$.

**Algorithm 1.** Recognition of malicious IPs with grid sorting

```
1: **Input:** risk profile $\Delta = (m, \delta)$
2: **Input:** Complete IP set $G$
3: Initialize set $G\_m$ to store the malicious IP addresses
4: Grid partitioning: $G = \{g_1, g_2, g_3, ..., g_n\}$
5: $G.sort()$ ▷ in decreasing order of feature $m$ and increasing order of other features
6: for $g$ in $G$ do
7:     $G\_m.add(g.items())$
8:     $val \longleftarrow \sum_{i \in g} f_m^{p^{(i)}}$
9:     $total\_eliminated \longleftarrow val + total\_eliminated$
10:    if $total\_eliminated >= \delta$ then
11:        Return $G\_m$
12:    end if
13: end for
```

Deriving the optimal solution of this optimization problem is expensive, especially when the network we are monitoring is at the ISP-level. Hence, we designed Algorithm 1 to get the near-optimal solution $G_m$ efficiently. Since the time complexity of sorting the IPs according to the risk degree is $O(nlogn)$, the algorithm conducts the grid partitioning on the searching space to accelerate the IP classification. Then, we need to eliminate IP addresses along the $m$ axis and minimize impacts on other features at the same time. With this grid configuration, we can always find a corner grid $g_m$ that has the largest value on feature $m$ but also has the smallest values on irrelevant features. The classifier considers the grid $g_m$ as the most suspicious grid and gives it the highest priority in classification. Afterwards, the algorithm sorts the remaining grids in the decreasing order of feature $m$ and increasing order of other features. Finally, the algorithm eliminates the IPs with the unit of a grid in such order until the overall traffic profile return to the benign area. Figure 3 shows an example of such procedure.

## 3.3 Adaptability

The proposed method features good adaptability, which means the users do not need to retrain the proposed model to fit it into a new network environment. We can use a variety of prior knowledge to evolve the model, making it even robust to the different environments.

Here, we assume the user will have some types of limited information about the new network environment as prior knowledge. Such information includes the measurement data or link bandwidth information about the network environment, some training samples for online learning, and the incomplete threshold values for DDoS detection. Any type of the above information can evolve the detection model and help the model adapt to the new environment.

**Network Traffic Measurement.** Assuming that we have the measurement data about the new network environment, we can normalize the KNN searching space from the trained environment to the new environment according to the traffic distributions of the two networks. The easiest way is using min-max normalization for the converting.

$$\begin{aligned} l &= max(D_{new}[:,i]) - min(D_{new}[:,i]) \\ \widehat{D}[:,i] &= l \cdot \frac{D[:,i] - min(D[:,i])}{max(D[:,i]) - min(D[:,i])} \end{aligned} \tag{4}$$

Equation 4 shows the converting procedure, where $D$ denotes the original training dataset and $D_{new}$ denotes the sampled traffic from the new network environment. By mapping the original training data to the new network environment, our approach is able to conduct DDoS detection without retraining nor re-collecting the training data.

**Online Learning.** If the traffic monitoring system can obtain labeled traffic with the system running, we can conduct online learning on the proposed detection model, thus making it gradually fit a new environment. The KNN algorithm does not require training, making it very suitable and efficient to conduct online learning. However, the KD-tree, along with the confirmed areas, needs to refresh to reflect new knowledge. We can control the program to update the classifier only during the idle time to reduce the performance impact on the detection system. Nevertheless, the time complexity of refreshing the model is only O(n).

**Incomplete Thresholds.** In some circumstances, users may know some incomplete threshold values or detection rules in a new network environment. They can then use the preliminary knowledge to build a decision tree and merge it with the trained classifier, a tree-like data structure. If the prior knowledge of the new environment contradicts with the trained detection model, the user can manually indicate the decision priority.

## 4 Evaluation

After implementing our approach, we trained the detection model with existing DDoS datasets and tested our approach in both simulation and real network environments to evaluate its performance. We also utilized FastNetMon [23], a commercial DDoS detection program, to conduct comparison tests with our approach. FastNetMon is a threshold-based DDoS detection program widely used in middle and small-sized enterprises due to its high efficiency and accuracy. Overall, the evaluation results show that our approach can accurately detect DDoS threats with a relatively low latency. Moreover, according to the result of real-world deployment, our approach can easily fit a new network environment without retraining.

### 4.1 Model Training

**Table 1.** Datasets for training and testing.

| Dataset name | Format | Attack type |
|---|---|---|
| DRAPA 2009 DDoS [3] | pcap | TCP SYN flood attack |
| CAIDA 2007 DDoS [2] | pcap | ICMP flood attack |
| FRGP NTP Flow Data [5] | Argus flows | NTP reflection attack |
| DDoS Chargen 2016 [6] | Flow-tools | UDP reflection and amplification attacks |

We picked some representative DDoS datasets from public repositories to train and test our approach. Table 1 shows the datasets we used and the types of attacks they contain. Those datasets can cover at least five types of DDoS attacks, which are TCP SYN flood attacks, ICMP flood attacks, UDP flood attacks, NTP reflection attacks, and UDP reflection and amplification attacks based on Chargen protocol. Thus, we trained five DDoS detection models respectively from the datasets. These models can collaborate to provide protections to the victim server.

The training datasets are in different formats, ranges from fine-grained pcap format to flow-level descriptions of the connections. Our approach works above the flow-level. Thus, to preprocess the data, we converted the original datasets to traffic profiles according to different detection models with the granularity of five seconds. We also sampled a small portion (around 10%) of data from the DDoS datasets as the testing datasets. Those selected datasets would not participate in the model training, but would appear in the testing phase.

As Table 2 shows, we selected five sets of features to train five different detectors. These detectors aimed at TCP SYN flood, ICMP flood, NTP reflection attack, UDP reflection attack, and UDP amplification attack, respectively. The most frequently used feature was the ratio of the inbound traffic volume to the outbound traffic volume. We found this feature plays an important role

**Table 2.** Features we use for detecting and classifying different categories of DDoS attacks.

| Attack type | Features we use |
|---|---|
| TCP SYN flood attack | #inbound TCP packets/#outbound TCP packets, #TCP bytes, #TCP packets, #SYN flags, #flows, #PSH flags |
| ICMP flood attack | #inbound ICMP packets/#outbound ICMP packets, #ICMP bytes, #ICMP packets, #echo requests, #echo replies |
| NTP reflection attack | #inbound NTP packets/#outbound NTP packets, #inbound NTP bytes/#outbound NTP bytes, #NTP bytes, #NTP packets |
| UDP reflection attack | #inbound UDP bytes/#outbound UDP bytes, #inbound UDP packets/#outbound UDP packets, #UDP bytes, #UDP packets |
| UDP amplification attack | #inbound UDP packets/#outbound UDP packets, #UDP packets, #UDP bytes |

in identifying the majority of DDoS attacks. Other than this, each attack has some specific features that are particularly useful for detection. For example, the number of NTP bytes and the number of NTP packets are essential to detect NTP reflection attacks, but not useful at all for detecting TCP SYN flood attacks.

## 4.2 Evaluation Under Simulation Environment

We first built a simulation environment to get convenient and efficient tests on the proposed method. To simulate the legitimate background traffic, we sampled network traffic from one router of the Front Range GigaPop (FRGP) [1], a well-known regional Internet service provider (ISP). During the evaluation, we converted the background traffic to traffic profiles and kept replaying them to the detection model. Simultaneously, we replayed the DDoS attack traffic, fine-tuning the traffic volume from a small size to an overwhelming size for mimicking the real scenario.

Figure 4 shows the performance of the DDoS threat detection. The proposed approach surpasses FastNetMon in both the accuracy score and recall score, which means it performs better at identifying DDoS threats. However, our approach is slightly inferior to FastNetMon at the precision score and false positive rate, which means our approach is more likely to label legitimate traffic as malicious. Fortunately, the explicable detection results can help network administrators to exclude false positive alarms quickly.

As for the traffic classification, we conducted access control on the malicious IP addresses reported by our approach and FastNetMon in the simulation environment, and then measured the network situation. Figure 5 shows the

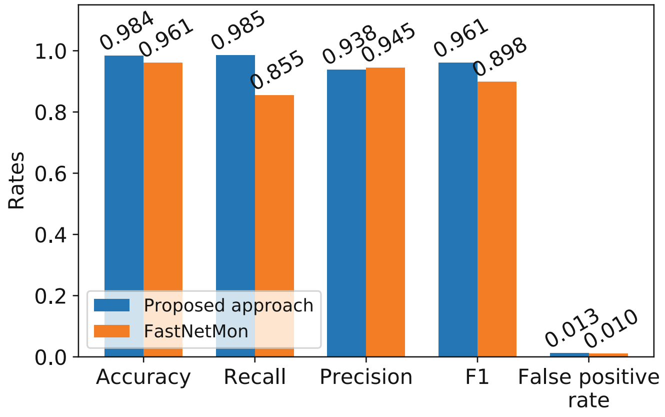

**Fig. 4.** Detection accuracy.

measurement results, where the y-axis indicates the number of packets. By mitigating all the traffic from the attackers classified by the two programs, we can see our approach can eliminate more malicious traffic than FastNetMon. The only drawback of our approach is that the classification will only be triggered when an attack is detected. Once the traffic profile returns to a legitimate area, our approach would not keep classifying malicious IP addresses until the overall traffic profile turns to hazardous again. Thus, there are periodic fluctuations on the number of packets for our approach in the figure.

## 4.3 Real-World Deployment

In addition to the evaluation under emulation environments, we deployed over approach on a 50 Gbps link in FRGP to test whether it can capture any DDoS traces in real world and evaluate its adaptability. We measured the network environment in the FRGP link and conducted profile normalization to map the training data to the new environment. We also used a newly developed flow-capture tool named FlowRide to keep streaming flow-level information to our approach every five seconds. A preprocessing program converts the flow records into the overall traffic profiles and IP-level traffic profiles. Each IP traffic profile is indexed by the feature numbers for prompt queries.

We deployed our DDoS detection approach from April 20th, 2020 to May 20th, 2020. During this period, the proposed approach detected six DDoS attacks. After manually verifying, five of the detection results are confirmed to be true positives, which contains two NTP attacks, a TCP SYN flood attack, a DNS reflection attack, and a bandwidth exhausting attack. The only fly in the ointment is that we missed a SYNACK DrDoS attack in the end of April. However, this attack is very rare and targeted the stateful firewall in front of a large AS. The training data did not contain the characteristics of the attack, making our approach difficult to capture it.

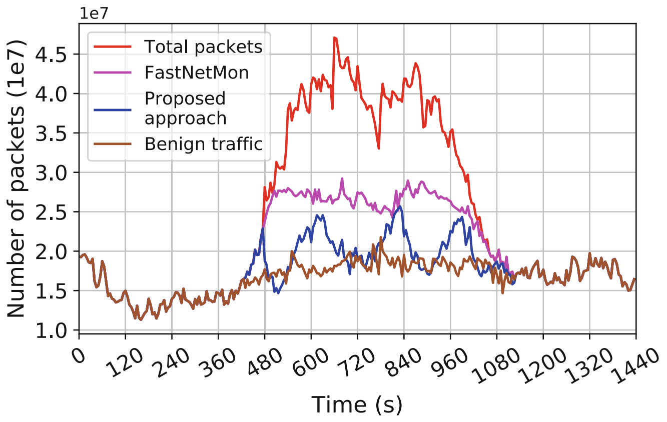

**Fig. 5.** Performance of traffic classification.

The real-world deployment also provides a good opportunity to test the delay with large throughput. We deployed our program on an Intel Xeon Silver 4116 processor with a RAM of 64 GB. Then, we measured the delay time of our approach and present the result in Fig. 6. From the figure we can see that the delay time is very short when there are no attacks happening. The detection model is similar to a decision tree and it will directly output the results without conducting any KNN queries if the traffic profile is in a good shape. Thus, the majority of time is spent on the data preprocessing when monitoring the traffic. When there is considerable DDoS attack traffic coming in, the delay time begins to increase. However, the detection and classification time is still less than the preprocessing time, costing merely two seconds for each. Besides, the total delay time does not increase significantly with the increase of the number of detection models. This is because only one model will be busy conducting queries at one time in typical cases. In conclusion, our approach is efficient when detecting and classifying DDoS traffic. With delays of around two seconds during idle time and six seconds during DDoS peak, our approach is able to give timely protections to the victim server.

## 5 Discussions

A primary contribution of this approach is to offer a learning-based DDoS detection solution that features good explainability and adaptability. However, this approach has the following limitations:

– As a learning-based approach, our method may not able to tackle zero-day DDoS attacks. Although the training data can be enhanced to cover more types of DDoS attacks to improve the capability of our approach, nonetheless, if the training does not include particular DDoS traffic, this approach probably will not be able to detect them.

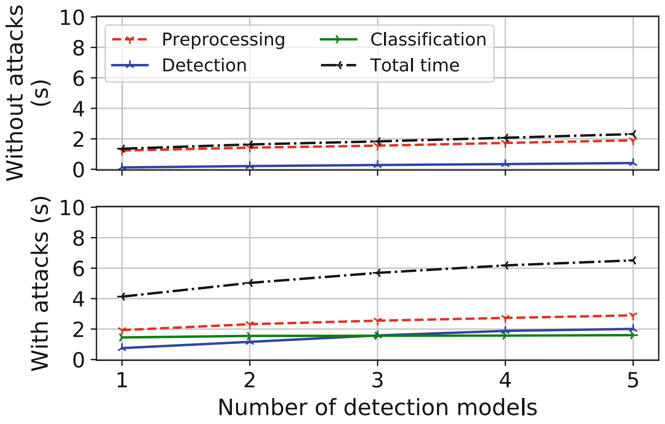

**Fig. 6.** Delay time (deployed on a 50 Gbps link).

– Due to the characteristic of KNN algorithm, our approach may be vulnerable to adversarial attacks. We can utilize some adversarial machine learning techniques such as data smoothing to fix this problem, but it will inevitably decrease the detection accuracy.

In addition, our approach faces several open issues as possible future working items:

– We can leverage some other explainable machine learning algorithms such as random forests to enhance our approach. Furthermore, it is meaningful to compare the current algorithm with random forests.
– More features may be explored to improve the accuracy for detecting certain complicated DDoS attacks.
– We can further enhance the evaluation of this work by testing the efficacy of our traffic classification algorithm.

## 6 Conclusions

This paper proposes a learning-based approach to detecting and classifying DDoS traffic. Compared with the existing approaches, the proposed method offers (1) explainability and (2) adaptability. With the KD tree and the modified KNN algorithm, this approach generates a tree-like classifier, which not only makes predictions fast but also provides the network administrator with a clear perspective of the network conditions. Furthermore, people can easily adapt the detection model to a different environment by using various prior knowledge *without* retraining the model from scratch. Benefiting from the grid sorting, the classification module can reduce collateral damage to the maximum extent and generate the results promptly.

We trained the detection model with representative DDoS datasets from public repositories in our simulation environment. We then evaluated this approach in both simulated environments and a real setting. The evaluation results show the efficacy and efficiency of this approach in both settings, as well as its adaptability from the small simulated environments to a real ISP setting. In addition, this approach comes with explainable information to indicate the intensity and category of the attack.

**Acknowledgments.** This project is the result of funding provided by the Science and Technology Directorate of the United States Department of Homeland Security under contract number D15PC00204. The views and conclusions contained herein are those of the authors and should not be interpreted necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Homeland Security or the US Government.

## References

1. Introduction of the front range GigaPOP (FRGP). https://www.frgp.net/intro.shtml
2. The CAIDA UCSD "DDoS attack 2007" dataset (2007). https://www.caida.org/data/passive/ddos-20070804_dataset.xml. CAIDA
3. DARPA 2009 intrusion detection dataset (Colorado State University) (2009). http://www.darpa2009.netsec.colostate.edu/
4. National cyber awareness system: Security tip - understanding Denial-of-Service Attacks (2009). https://www.us-cert.gov/ncas/tips/ST04-015. US-Cert
5. FRGP NTP flow data - NTP reflection attack (2014). https://www.impactcybertrust.org/dataset_view?idDataset=776. University of Southern California-Information Sciences Institute
6. DDoS Chargen 2016 dataset - Internet traffic data containing a DDoS attack based on UDP Chargen protocol (2016). https://www.impactcybertrust.org/dataset_view?idDataset=693. Merit Network Inc
7. Cisco annual internet report (2018–2023) white paper (2020). https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report
8. Barati, M., Abdullah, A., Udzir, N.I., Mahmod, R., Mustapha, N.: Distributed denial of service detection using hybrid machine learning technique. In: International Symposium on Biometrics and Security Technologies, pp. 268–273. IEEE (2014)
9. Bhaya, W., EbadyManaa, M.: DDoS attack detection approach using an efficient cluster analysis in large data scale. In: Annual Conference on New Trends in Information & Communications Technology Applications, pp. 168–173. IEEE (2017)
10. Bhuyan, M.H., Kashyap, H.J., Bhattacharyya, D.K., Kalita, J.K.: Detecting distributed denial of service attacks: methods, tools and future directions. Comput. J. **57**(4), 537–556 (2014)
11. Chen, C.L.: A new detection method for distributed denial-of-service attack traffic based on statistical test. J. Univ. Comput. Sci. **15**(2), 488–504 (2009)
12. Claise, B.: Cisco systems netflow services export version 9. RFC 3954 (2004)
13. Doshi, R., Apthorpe, N., Feamster, N.: Machine learning DDoS detection for consumer Internet of Things devices. In: IEEE Security and Privacy Workshops, pp. 29–35 (2018)
14. Feng, Y., Li, J.: Towards explicable and adaptive DDoS traffic classification. In: The 21st Passive and Active Measurement Conference - Poster, March 2020

15. Feng, Y., Li, J., Nguyen, T.: Application-layer DDoS defense with reinforcement learning. In: IEEE/ACM International Symposium on Quality of Service (2020)
16. Friedman, J.H.: On bias, variance, 0/1-loss, and the curse-of-dimensionality. Data Min. Knowl. Discov. **1**(1), 55–77 (1997). https://doi.org/10.1023/A:1009778005914
17. He, Z., Zhang, T., Lee, R.B.: Machine learning based DDoS attack detection from source side in cloud. In: The 4th International Conference on Cyber Security and Cloud Computing (CSCloud), pp. 114–120 (2017)
18. Kokila, R., Selvi, S.T., Govindarajan, K.: DDoS detection and analysis in SDN-based environment using support vector machine classifier. In: The Sixth International Conference on Advanced Computing, pp. 205–210. IEEE (2014)
19. Lee, K., Kim, J., Kwon, K.H., Han, Y., Kim, S.: DDoS attack detection method using cluster analysis. Expert Syst. Appl. **34**(3), 1659–1665 (2008)
20. Limwiwatkul, L., Rungsawang, A.: Distributed denial of service detection using TCP/IP header and traffic measurement analysis. IEEE International Symposium on Communications and Information Technology, vol. 1, pp. 605–610 (2004)
21. Lu, K., Wu, D., Fan, J., Todorovic, S., Nucci, A.: Robust and efficient detection of DDoS attacks for large-scale internet. Comput. Netw. **51**(18), 5036–5056 (2007)
22. Mirkovic, J., Reiher, P.: D-WARD: a source-end defense against flooding denial-of-service attacks. IEEE Trans. Dependable Secure Comput. **2**(3), 216–232 (2005)
23. Odintsov, P.: FastNetMon-very fast DDoS analyzer with sflow/netflow/mirror support. https://github.com/pavel-odintsov/fastnetmon/
24. Panchen, S., Phaal, P., McKee, N.: InMon corporation's sFlow: a method for monitoring traffic in switched and routed networks (2001)
25. Seo, J., Lee, C., Shon, T., Cho, K.-H., Moon, J.: A new DDoS detection model using multiple SVMs and TRA. In: Enokido, T., Yan, L., Xiao, B., Kim, D., Dai, Y., Yang, L.T. (eds.) EUC 2005. LNCS, vol. 3823, pp. 976–985. Springer, Heidelberg (2005). https://doi.org/10.1007/11596042_100
26. Suresh, M., Anitha, R.: Evaluating machine learning algorithms for detecting DDoS attacks. In: Wyld, D.C., Wozniak, M., Chaki, N., Meghanathan, N., Nagamalai, D. (eds.) CNSA 2011. CCIS, vol. 196, pp. 441–452. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22540-6_42
27. Thomas, R., Mark, B., Johnson, T., Croall, J.: NetBouncer: client-legitimacy-based high-performance DDoS filtering. In: Proceedings DARPA Information Survivability Conference and Exposition, vol. 1, pp. 14–25. IEEE (2003)
28. Wang, J., Phan, R.C.W., Whitley, J.N., Parish, D.J.: Augmented attack tree modeling of distributed denial of services and tree based attack detection method. In: The 10th IEEE International Conference on Computer and Information Technology, pp. 1009–1014 (2010)
29. Yuan, X., Li, C., Li, X.: DeepDefense: identifying DDoS attack via deep learning. In: IEEE International Conference on Smart Computing, pp. 1–8 (2017)
30. Zekri, M., El Kafhali, S., Aboutabit, N., Saadi, Y.: DDoS attack detection using machine learning techniques in cloud computing environments. In: The 3rd International Conference of Cloud Computing Technologies and Applications, pp. 1–7. IEEE (2017)
31. Zhang, G., Jiang, S., Wei, G., Guan, Q.: A prediction-based detection algorithm against distributed denial-of-service attacks. In: International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly, pp. 106–110 (2009)
32. Zi, L., Yearwood, J., Wu, X.W.: Adaptive clustering with feature ranking for DDoS attacks detection. In: The Fourth International Conference on Network and System Security, pp. 281–286. IEEE (2010)