Contents lists available at ScienceDirect

Computer Networks

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

The hybrid technique for DDoS detection with supervised learning algorithms

Soodeh Hosseini[a,b,*], Mehrdad Azizi[a]

[a] Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran
[b] Mahani Mathematical Research Center, Shahid Bahonar University of Kerman, Kerman, Iran

A R T I C L E I N F O

Article history:
Received 5 August 2018
Revised 13 January 2019
Accepted 25 April 2019
Available online 25 April 2019

Keywords:
DDoS
Machine learning
Incremental learning
Hybrid mechanism

A B S T R A C T

Distributed denial of service (DDoS) is still one of the main threats of the online services. Attackers are able to run DDoS with simple steps and high efficiency in order to prevent or slow down users' access to services. In this paper, we propose a novel hybrid framework based on data stream approach for detecting DDoS attack with incremental learning. We use a technique which divides the computational load between client and proxy sides based on their resource to organize the task with high speed. Client side contains three steps, first is the data collecting of the client system, second is the feature extraction based on forward feature selection for each algorithm, and the divergence test. Consequently, if divergence got bigger than a threshold, the attack is detected otherwise data processed to the proxy side. We use the naïve Bayes, random forest, decision tree, multilayer perceptron (MLP), and k-nearest neighbors (K-NN) on the proxy side to make better results. Different attacks have their specific behavior, and because of different selected features for each algorithm, the appropriate performance for detecting attacks and more ability to distinguish new attack types is achieved. The results show that the random forest produces better results among other mentioned algorithms.

1. Introduction

Distributed denial of service (DDoS) is a common and critical web service attack in recent years due to its simple operation and high efficiency. For example, a DDoS attack includes such a power which is able to disconnect a country from the internet. Here the DDoS attacks is considered as a cyber-warfare tactics [1]. The purpose of the DDoS attack is to deny legitimate users' access to services by exhausting the hardware resource or bandwidth which means servers lose their availability and is an important part of the security of any services. The Computer Incident Advisory Capability (CIAC), reported the first DDoS attack occurrence. This type of attack consist of a previous version which is Denial of service (DoS) and is known since 1980 [2]. The difference between these two types of attacks is in the number of sources which is used in a Dos attack, included a single source, but in DDoS, there are botnets that consist of several compromised machines called zombies which are controlled by bot master as the attacker. There are many tools to generate DDoS attacks such as Trinity, low orbit ion cannon, tribal flood network, and mstream, trinoo. These tools are different in their architecture, type of flooding attack, and also the method that is used for DDoS attack and several other aspects [3].

Different definitions have been proposed for machine learning such as machine learning addresses and how to build computers that improve automatically through experience [4] or machine learning is about predicting the future based on the past that is questioned. If the time into the past and future is divided, in the past it is learned from training data then in the future, will have be tested data and will be created a model or predictor which can be used for the purpose of predicting next steps [5]. Machine learning is rapidly growing in many fields such as science, technology, marketing, education, healthcare, and many other fields. Several applications of machine learning are natural language processing (NLP), speech processing, autonomous vehicle control, robotics, image processing and computer vision [4]. Machine learning technique in cyber security is helpful by recommending the proper decision for analysis and even doing the proper action automatically. Some techniques which are available include artificial neural networks, association rules, and fuzzy association rules, the Bayesian network, clustering, decision trees, ensemble learning, evolutionary computation, hidden Markov models, naïve Bayes and support vector machine [6]. The researchers' approach in this study includes using some algorithms together and takes the best action based on the presented target and algorithm result.

* Corresponding author.
E-mail address: so_hosseini@uk.ac.ir (S. Hosseini).

https://doi.org/10.1016/j.comnet.2019.04.027
1389-1286/© 2019 Elsevier B.V. All rights reserved.

The given implementation is based on the open source analytics platform KNIME, a modular environment that allows interactive execution and simple visual assembly of workflows. In this tool, it can be simply implemented an algorithm, read data in various formats and manipulate them, visualize results and many other tasks are at hand [7].

The fast-growing usage of online services and data generation in the world is led to the emergence of a significant problem which is the way to handle big data and protect online services. There are many significant and useful wisdom and information in the data. Therefore, the huge profit can get from data cannot be ignored [8].

Most of data is generated in a sequential or stream, therefore this type of data should be considered. There are multiple techniques available for handling this data, and will be described further in the following sections. Now with increasingly exponential data in the world, the technique when facing an online threat such as a DDoS attack must be improved. Also, both big data and stream data topics in this work should be considered.

Two main contributions of this work are given. The first major aim is to divide and balance the process between client and proxy to derive a better result in a specified time due to the resources limitless. The goal that is intended to be achieved is no overweight on both sides especially the client side. Secondly, in order to prevent the client to continue the attack in the first place by detecting the attack as soon as possible, either on the client side or the proxy side.

In this paper, a new hybrid framework is introduced based on data stream approach for DDoS attack detecting with a technique which divides the computational load between the client and proxy side.

Furthermore, the presented work reduces process time and process cost on intrusion detection system (IDS) by giving some of the processes to the client side. Using a number of algorithms together can resolve the weaknesses of the others and give a better detection rate to IDS. Algorithms have different process procedures; therefore, this framework is able to handle new attack types better than any other presented framework.

Moreover, in Section 2, the main study concepts are shortly explained. In Section 3 an overview of some related works is given. The proposed framework will be introduced in Section 4, and experimental results are followed in Section 5. The last section includes the results of this paper.

2. Background

In this section, the concepts of DDoS attacks and learning algorithms which are used in this work are discussed briefly.

2.1. DDoS attack type classification

Furthermore In the following section a description of some of the DDoS attack types is given [1,9]:

2.1.1. UDP flood attack

User datagram protocol (UDP) is a network transmission protocol for transporting data in the network with more speed and without any permission. Therefore, the attacker can send a packet with a larger size than the average, causing an attack. This type of attack can diminish hosts, also, is able to exhaust networks.

2.1.2. ICMP (Ping) flood

Similar to the UDP, internet control message protocol (ICMP) or ping, intends to send the packet in the fastest way possible without the need for any reply or confirmation. Attackers of this type will send numerous false requests to the victim. Hence network traffic is exhausted.

2.1.3. SYN flood

In this technique, attackers continuously send packets to the server with fake sender address in order to prevent the server to close the connection. Therefore, the connection remains online to increase the number of connections and the other system cannot connect to the server because the connections are saturated by the attackers. The aim of this attack is to consume as much as possible of the victim's resources.

2.1.4. Bandwidth attack

The attacker will overload traffic to the victim's server for shutting down the server, slowing down the performance, or even damaging the infrastructure. The attack which is explained before can be defined as an example of this type.

2.2. DDoS detection and defense classification

According to the [10,11] references, detection and defense techniques are divided into four types which are described in the following:

2.2.1. Source side defense mechanism

This mechanism is deployed near the source of the attack to prevent users from being part of a DDoS attack by analyzing outgoing traffic and setting traffic limitation in order to reduce or defeat attack with the lowest damage on rightful traffic. This mechanism can also be deployed at the edge router of source local network or at the access router of an autonomous system (AS) which is connected to the edge routers. Some examples of this type of mechanism are given in the following:

1. Ingress/Egress filtering at the sources' edge/routers.
2. D-WARD [12].
3. Multi-level tree for online packet statistics (MULTOPS).
4. Mananet's reverse firewall, etc.

2.2.2. Network base defense mechanism

This type of mechanism is deployed in the network and any router. It can either act as a defender by itself for users and or it is able to perform the appropriate actions for responding and defeating the attacker. It should be mentioned that the focus of this paper is on this type of mechanism. These actions can set filter or rate limit the traffic. This mechanism helps to find the source of attack by cooperative operation between networks adapters. In this mechanism, networks adapters received legitimate and aggregated malicious traffic data, therefore filtering is not a suitable option. Finally, better option is set to rate limit on the traffic. For instance, some mechanisms are as follows:

1. Route-based packet filtering.
2. Detecting and filtering malicious routers.

2.2.3. Victim side defense mechanism

In victim-end, almost the whole detection and response mechanism has been done only in the destination of the attack, therefore, first of all, the attack should be detected and then defense strategy on the server by filtering or setting a limitation on malicious traffic must be applied. Because of deployment point, the victim traffic precisely is observed and is detected anomalies, since attack traffic arriving at the server causes the service to get slow. However, with all advantage in this mechanism because of huge DDoS attack traffic, the victim sources will be exhausted to defeat the attack, so will be vulnerable [13]. Some destination-based DDoS defense mechanisms in the following:

1. IP trace back mechanisms.
2. Management information base (MIB).
3. Packet marking and filtering mechanisms and etc.

2.2.4. Hybrid defense mechanism

In most of the mechanism which are described before, there is no collaboration between any of components. The hybrid defense mechanism is introduced to resolve this problem and make the capability in order to create cooperation between the components. Deployment point is able to have several locations such as victim, intermediate networks, and source. This strategy combines the previous strategies, which are called centralized. Using this mechanism, the strength of other mechanisms together is used. For example, network base on this mechanism is better in rate limit on traffic, but victim side can better distinguish between legitimate traffic and malicious traffic.

2.3. Learning methods

In this work, two kinds of data which are big data, the data generated in the networks that are extremely high and stream data which means the data generated during the time are presented. The machine learning from these two types in the following will be described.

2.3.1. Big data learning

In recent years big data has grown in most of the fields such as science, engineering and many various other fields. Using the term big data, with referring to the data set, that includes a large amount of data. Based on the report from international data corporation (IDC) in 2011, 1.8 ZB or 1021 B data exists throughout the world, which is about nine times bigger than five years than before. Growing data will be approached double at least for every two years. Many companies engage in the term of big data, for example, Google, Facebook, TaoBao and many others example [14]. There is a lot of significant knowledge and information in data; thus extracting information's calls for new learning techniques to handle the challenge of data [8]. In this work, the system's activity and system's events to detect intrusion are analyzed. System's activity and system's events can generate big data. It is possible to gather data from multiple systems and process data to extract anomalies and malicious activities. After extracting information from big data, effective models to prevent and detect attack activity can be created [15]. Big data learning includes problems such as [8]:

1. Large scale of data.
2. Different types of data.
3. High speed of streaming data.
4. Unc1ertain and incomplete data.
5. Data with low-value density and meaning diversity.

2.3.2. Stream data learning

Stream data is a kind of data, which continuously arrives and every data is visible one time.

In the past, hypothesis based on that concept, all available examples was accessible, load into memory was feasible. However, in recent years, the examples that are generating data during the time such as user modeling in social networks, monitoring of community network, web mining and etc. [16] are observable. In the other hand, the size of data exponentially grows up so it does not have the ability to load into the memory. This problem makes use of a new mechanism. Here are two strategies available for handling this challenge:

1. The parallel processing: the algorithm into the little and separate part to reduce computing time is divided.
2. The incremental processing: a one-pass algorithm to create a model and update that each time read a single data is implemented.

In stream data learning some problems are as follows:

1. Processing great size of streaming events like predict, act, filter, etc.
2. Scalability and performance when the size and complexity of data are increased.
3. Analytics like real-time data discovery and monitoring, process on arriving query continuously, alarm and responses automatically, etc.

There are many tools for handling streaming data, for example, apache storm, spark streaming, amazon kinesis, and several other tools [17].

2.3.3. Incremental learning

Incremental learning is a type of learning, which is based on the fact that if a new example and hypothesis f_i is given, the hypothesis f_{i+1} without learning on all of previous data or hypothesis can be generated. In this type, the algorithm must be faster than other algorithms similar to batch learning algorithm. To achieve this goal, most incremental algorithms are read data for a single time. Therefore, the time efficiency increases and can process more data in a short time.

In addition the properties which is mentioned before other properties such as, need fixing and low time for processing each example, reading data based on their order and just in one time, constant consuming memory but no matter how many examples processed and predict in anytime are expected [16].

3. Related work

Until now, some of the basic definitions of presented work are explained. This section will be a summary of the related works. In the following, some of the researches which is done in recent years is shown in Table 1. This table is based on three columns, objective, deployment and remarks. The objective is the main aim of each work, which is mostly the attack detection except one of works. Deployment, which is described above, includes four types, source side, network base, victim side, and hybrid. At remarks column, the summary of the points of each work are explained.

4. The proposed framework

The proposed approach framework is a hybrid machine learning mechanism for detecting and defensing against DDoS attack. The given framework is based on two sides, client side and proxy side. Because of the limited resource on both sides, the process between them is divided.

There are many papers for detecting DDoS attack using machine learning algorithm. Each of papers uses one or multiple algorithms separately to detect and compare the results with each other [23,24,28,29]. Here it is tried to use multiple algorithms together and benefit from all of algorithms properties simultaneous [30]. Also, in the framework, a determiner is used which helps improving results. In determiner, there are several rules to achieve better results based on classifiers under the particular situation and system admin can define rules.

In Fig. 1 the given proposed framework is shown. Detection process starts on the client side and after some preliminary step data are compared with the attack profile database and if it is not attacked, data proceed to the proxy side. Finally, if data detected as normal, it proceeds to the server.

4.1. Dataset

The first step of each learning is data gathering. Appropriate data is helpful to have a better result and designing framework

Table 1
Aattack detection study in literature.

Reference	Objective	Deployment	Remarks
Sekar et al. [18]	Attack detection	Source side	Design triggered framework, a multi-stage framework with high accuracy and scalability property
Rahmani et al. [19]	Attack detection	Victim side	A statistical approach based on the network anomaly and joint entropy using joint entropy analysis of multiple traffic distributions
Gupta et al. [20]	Number of zombies identification	Victim side	Approximate number of zombies in a DDoS attack by ANN
François et al. [21]	Attack detection	Source side	An early detection technique for DDoS flooding attack which supports incremental deployment in real network such as internet service providers (ISPs)
Barati et al. [22]	Attack detection	Intermediate network	Detecting DDoS attack by ANN and feature selection via GA algorithm.
Singh and Panda [9]	Attack detection	Intermediate network	Anomaly and signature-based detection from high-speed link in an efficient data streaming fashion
Fadlil et al. [23]	Attack detection	Source side	Network traffic activity was statistically analyzed using Gaussian naïve Bayes method
Kim and Lee [24]	Attack detection	Source side	Proposed some attributes and a method for detecting a variant of DDoS attack on the client side using SVM.
Yusof et al. [25]	Attack detection	Intermediate network	Proposed a technique called PTA-SVM (Packet Threshold Algorithm with SVM) to detect DDoS attacks in the first place then detected the type of attack in four types of attack which are TCP SYN flood, UDP flood, Ping of Death and Smurf with 99.1 accuracy and positive rates 1.11.
Zhou et al. [26]	Attack detection	Intermediate network	Proposed online DDoS detection system with three main modules: a collector for capture packet from the network implemented by Jpcap. Messaging system with Kafka to transmit data from collectors to stream processor and stream processor is a cluster running spark streaming. Also They evaluated their framework with naïve Bayes, logistic regression, decision tree and achieve accuracy up to 99.3 even for internet traffic.
Behal et al. [13]	Attack detection	Intermediate network	Proposed an ISP level, flexible and automatic model based on collaborative with the nearest point of presence routers to distribute computational and storage complexity.
Idhammad et al. [27]	Attack detection	Intermediate network	Proposed semi-supervised method based on entropy, co-clustering information gain ratio, and the extra-trees ensemble classifiers.
Mallikarjunan et al. [28]	Attack detection	Victim side	First of all, create a dataset in five steps and train naïve Bayes algorithm as the main method, then random forest and J48 for comparing results. Finally, they find out naïve Bayes has better results.

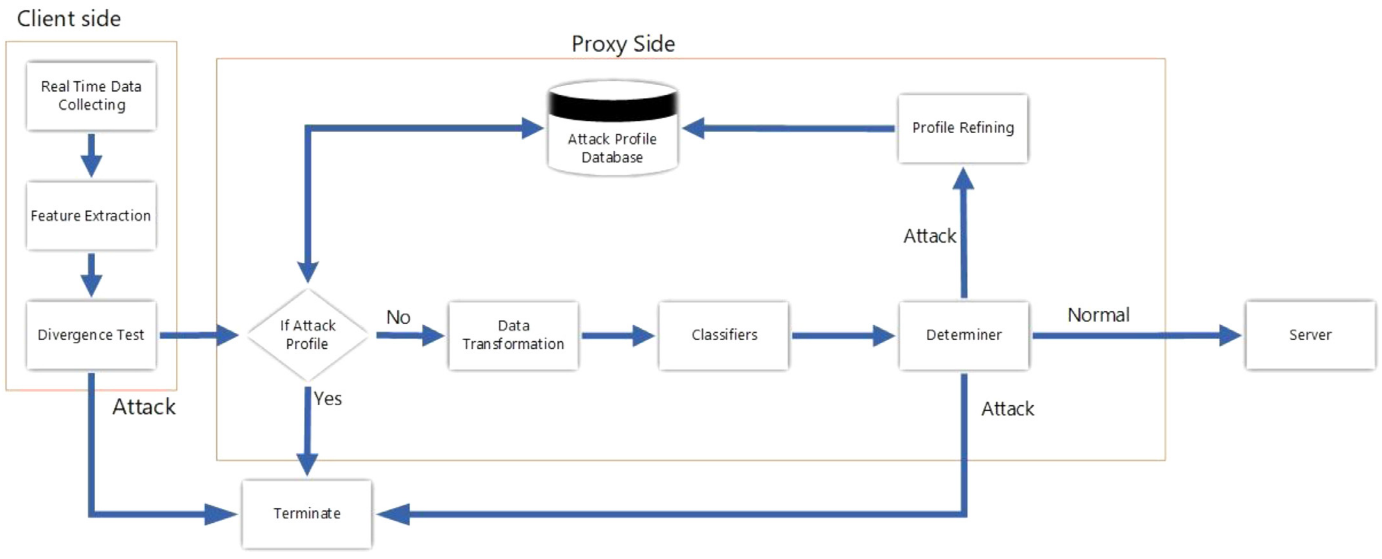


Fig. 1. The proposed framework.

efficiently. The presented framework result is based on two distinct datasets.

The first dataset is NSL-KDD dataset [31]. NSL-KDD is an improved dataset from KDDCUP'99, which solves several problems of KDDCUP'99 and prepares new a dataset with the selected records from KDD dataset with no previous problem to exist anymore. This means despite the problems of NSL-KDD, is a proper dataset for researchers [32]. NSL-KDD contains not only DDoS attack but also different types of attack. In the first place contains 125,974 records for train data and 22,544 records for test data with 43 attributes. A filter to distinct DDoS attack from other attacks and decrease records to 113,270 and 17,168 for train and test data is applied. Moreover, it is assumed that all type of DDoS attack in the dataset is same and renamed all type to DDoS, therefore two classes "DDoS" and "normal" are given.

The second dataset is a dataset which was provided by Alkasassbeh et al. [33]. In this dataset, the modern attacks like SIDDOS and HTTP flood is presented. Dataset gathering is done by authors in six-step. First of all, the network traffic is generated then is collected and tested after preprocessing on data. When preprocessing is done, the features extraction is started and then is calculated statistical measurement and finally the dataset will be ready. This dataset includes 2,160,668 records with 28 attributes. In dataset, five classes are defined, UDP-Flood, Smurf, SIDDOS, HTTP-FLOOD that are four type of DDoS attack and a class normal. The records into two classes "DDoS" and "normal" like NSL-KDD dataset are divided.

Before any operation on datasets, data are transformed into numerical and then normalize.

4.2. Offline side

Before deployment the framework some data manipulations is done and then train the classifiers. These steps are helpful to have better performance in client and proxy side due to the time-consuming of these steps. In the previous section, it is mentioned to some of steps, after that the forward selection is used to find the best subset of features for train of each algorithm separately. Forward selection starts with an empty subset of features and in every iteration adds a feature to improve the performance model. This iterative work continues until add any feature which does not improve performance. The pseudocode is presented as follows:

```
//input: our datasets and classifiers
//output: selected features and trained classifiers
1. dataset = [NSL-KDD, IntroIn33]
2. classifiers = [naïve Bayes, random forest, decision tree, MLP, K-NN]
3. for each ds in dataset:
4. for each cls in classifiers:
5. selected-features-ds-cls = forwad_feature_selection(ds, cls)
6. for each ds in dataset:
for each cls in classifiers:
8. trained-cls = train_classifiers(selected-features-ds-cls, cls, ds)
```

4.3. Client side

In this section the beginning of the framework on the client side will be done. In client side, because of less resource, the simple process such as data gathering from system activity and system events and then preprocessing for a-divergence test and finally, compare to presented models is given. If the result of the test is more than the threshold, the connection or any appropriate action to prevent the attack as soon as possible is terminated.

Pseudocode of client side is as follow:

```
//input: selected features in offline mode
//output: two type of packet attack or normal and suspicious
1. data = real_time_data_collect()
2. values = data_extraction(data, selected-features)
3. flag = divergence_test(values)
4. if (flag == "Attack")
5. terminate connection
6. else
7. process to proxy side
```

4.4. Proxy side

Proxy side architecture is shown in Fig. 2. After the client-side process, first, the incoming data with attack profile database due to avoid computational overhead is checked and if the results do not match any profile, the result is given to proxy side with an appropriate form, which can process machine learning algorithms on data. The naïve Bayes, random forest, decision tree, MLP and K-NN algorithms on data are processed, and then an algorithms determiner is given to the results which decides based on its configured policy and provides the better result. The approach based on incremental learning with every single input data, the profile is updated. On the other hand, here is a database of profile which prevents congestion of profiles. In continue, the database with the new results is refined.

In the following pseudocode of proxy side and its design is proposed

```
//input : packets from client side, trained classifiers from offline side
//output : two type of packet attack or
1. classifiers = [naïve Bayes, random forest, decision tree, MLP, K-NN]
2. data = received data from client
3. flag = check_profile(data, attack-profile-database)
4. if (flag == "Attack")
5. terminate connection
6. data = transform(data)
7. i = 0
8. for each cls in classifiers:
9. ans[i] = classifier(data, cls)
10. i = i + 1
11. TypeFlag, ProfileFlag = determiner(ans)
12. if (TypeFlag == "Attack")
13. terminate connection
14. if (ProfileFlag == "true")
15. update_database(data, attack-profile-database)
16. else
17. process to server
```

5. Experimental results

In this section the experiments implementation was carried out by analytics platform KNIME. In Fig. 3 the snapshot from KNIME workflow is considered, which is designed for implementation of the experiment. The machine learning algorithms is operated in batch mode and in future work and the stream data mode will be studied.

Experiment implementation consist of three sections, first is the data preprocessing to transform data into presented preferred form. In order to reduce processing cost on the proxy side and also to improve detection the feature selection is performed. Results are shown in the last section with different measurements such as recall, precision, and f-measure.

For fine-tune parameters, it is tried to find the best parameters for algorithms practically. The MLP trainer is configured by one hidden layer with 19 nodes for NSL-KDD and one hidden layer with eight nodes for the introduced dataset in reference [33]. For the random forest, the information gain ratio is used for split criterion and set the number of model 29 and 50 for NSL-KDD and the introduced dataset in reference [33] respectively. The decision tree trained with quality measures Gini index and Minimum Description Length (MDL) pruning method [34]. KNN trained with *k* equal to 11 for NSL-KDD and 7 for the other dataset. For NSL-KDD, closer neighbors have a greater influence on the resulting class than the other ones.

5.1. Data preprocessing

In the first step of implementation, the preprocessing on our datasets is exerted. Before common processes, for NSL-KDD the DDoS attack records is selected. Then for both of presented datasets, the string type attributes in data is changed to the integer type and normalized data with the min-max algorithm. In the next step, the different type of DDoS attack is renamed, which are defined in the dataset to two main classes, "DDoS" and "Normal".

5.2. Feature selection

In KNIME feature selection in a Meta node which name is forward feature selection can be performed. In this node, the dataset for input is given, and after processing, a table with attributes number and our accuracy when selecting each group of them is provided. A snapshot of feature selection result in Fig. 4 is shown which rows are sorted based on their descending accuracy and next to each percent of accuracy, and the number of features which are selected for the accuracy. On the right side of Fig. 4, the selected features for selected accuracy are marked with a blue line.

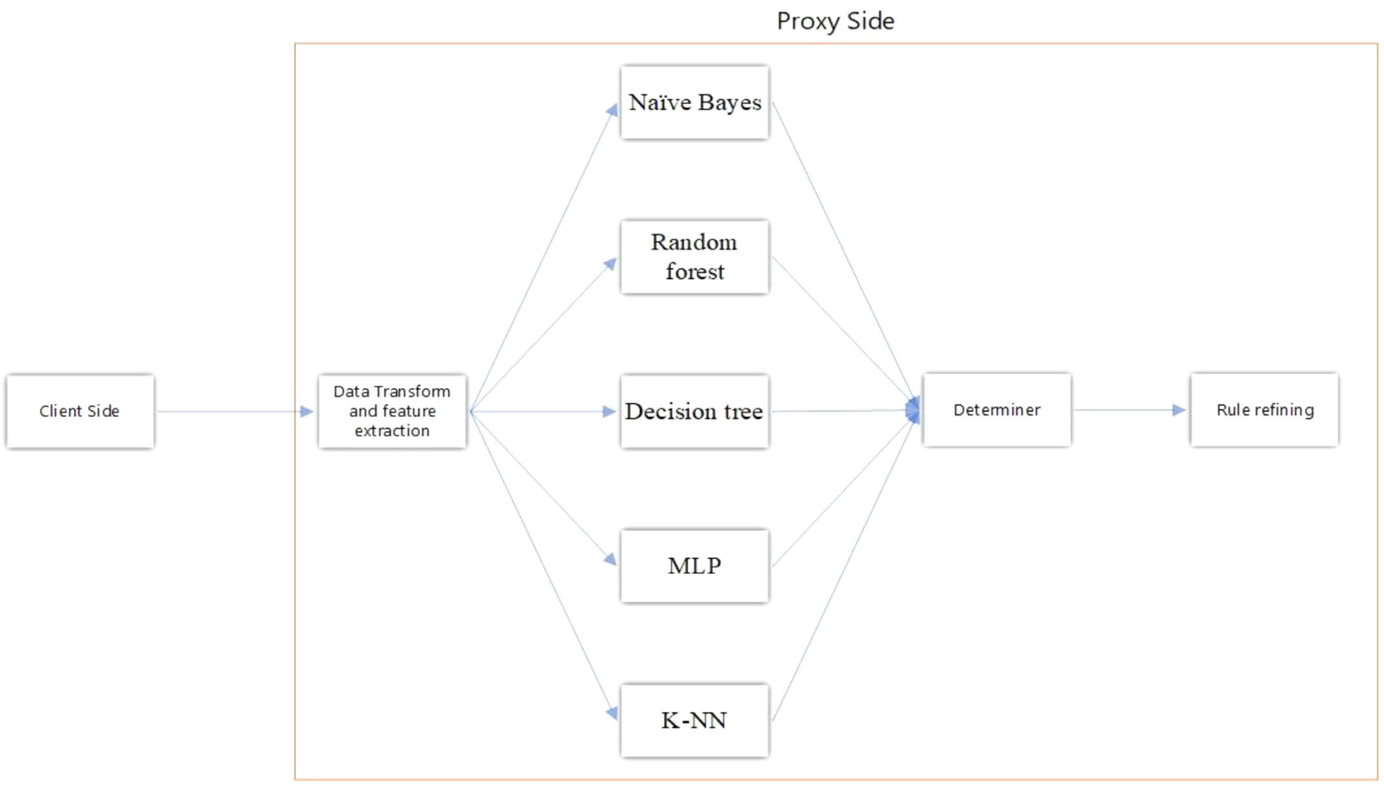


Fig. 2. Design of proxy side algorithm.

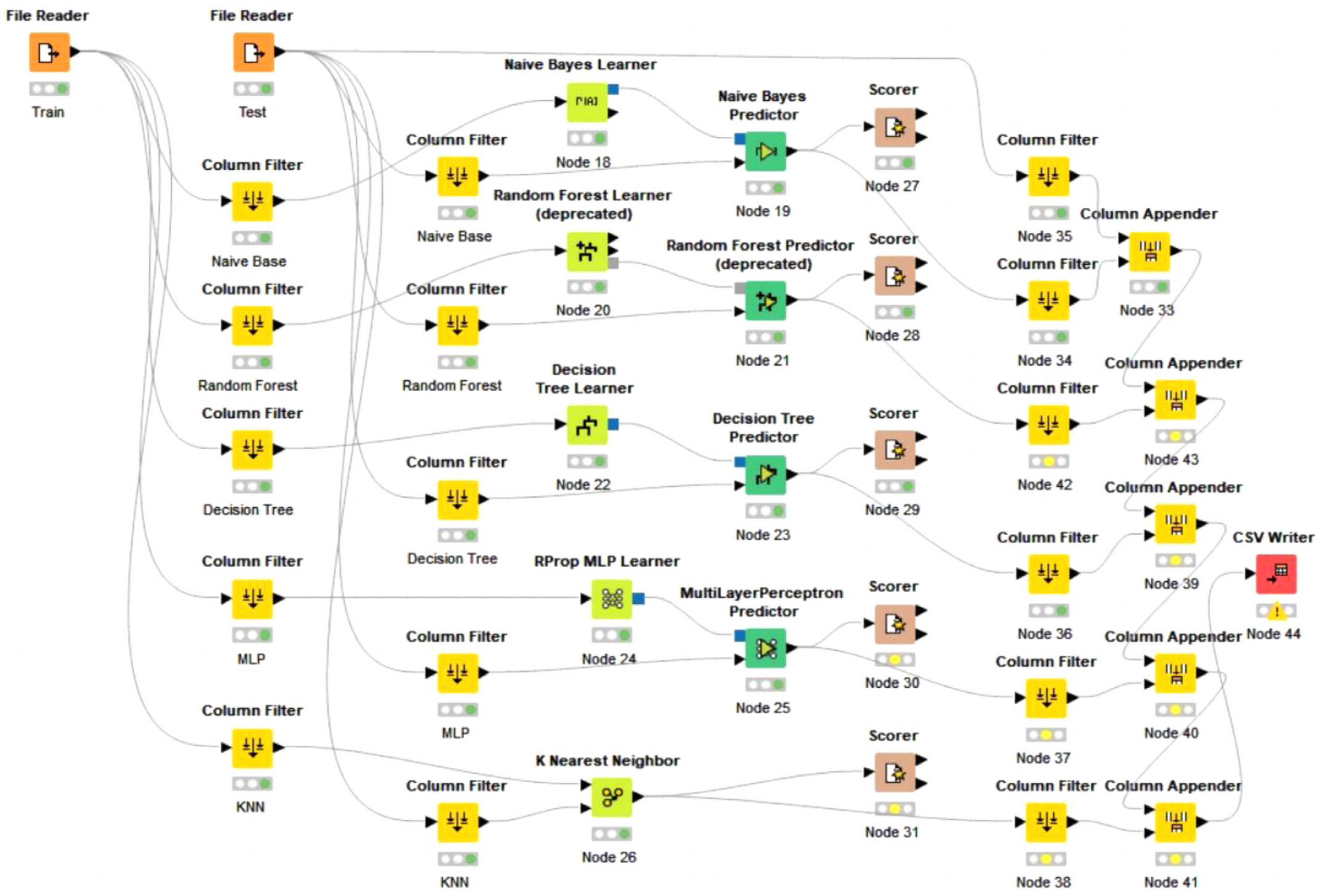
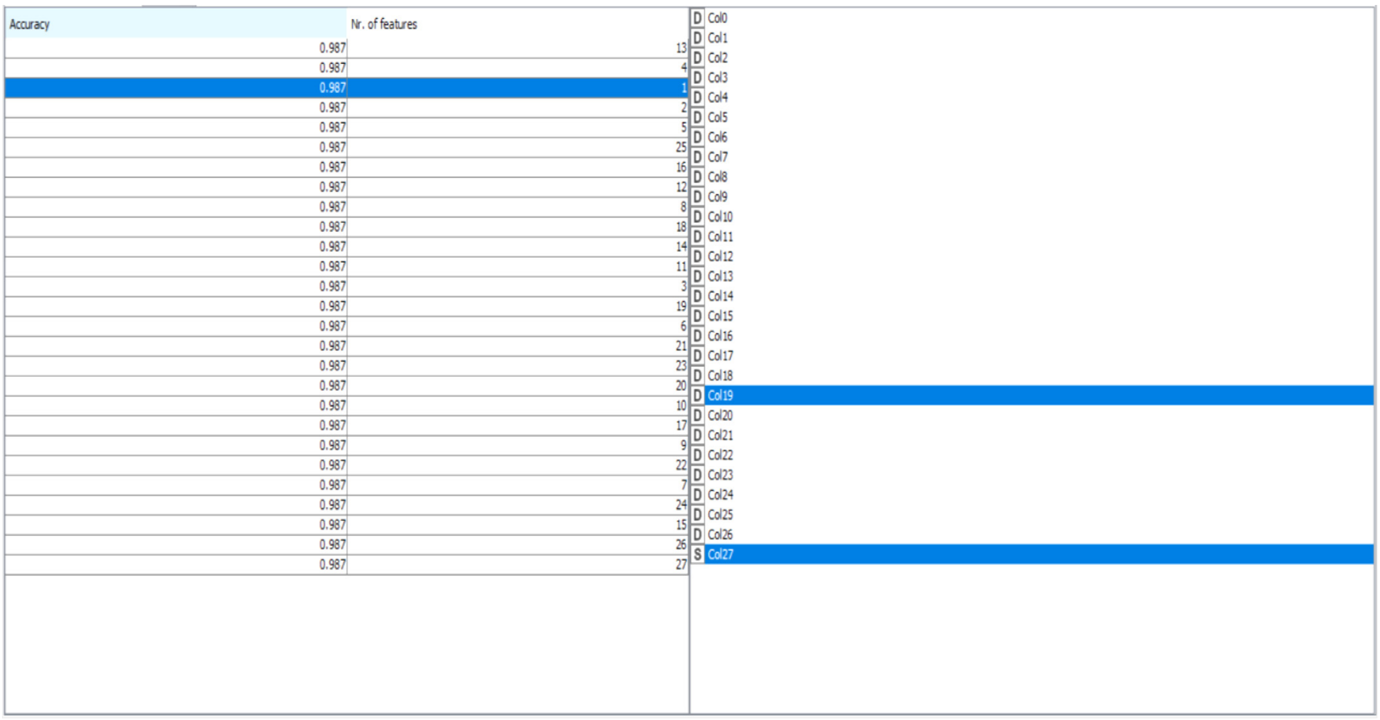


Fig. 3. KNIME workflow diagram.



Accuracy	Nr. of features
0.987	13
0.987	4
0.987	1
0.987	2
0.987	5
0.987	25
0.987	16
0.987	12
0.987	8
0.987	18
0.987	14
0.987	11
0.987	3
0.987	19
0.987	6
0.987	21
0.987	23
0.987	20
0.987	10
0.987	17
0.987	9
0.987	22
0.987	7
0.987	24
0.987	15
0.987	26
0.987	27

```
D Col0
D Col1
D Col2
D Col3
D Col4
D Col5
D Col6
D Col7
D Col8
D Col9
D Col10
D Col11
D Col12
D Col13
D Col14
D Col15
D Col16
D Col17
D Col18
D Col19
D Col20
D Col21
D Col22
D Col23
D Col24
D Col25
D Col26
S Col27
```

Fig. 4. KNIME forward feature selection results.

Table 2
Feature selection.

Algorithm	NSL-KDD		The introduced dataset in [33]	
	Features subset	Accuracy	Features subset	Accuracy
Naïve Bayes	1,3,7,21,27,28,31,34,35,38,41, 42	93.1	1,2,9,10,16,19,20,25,27	98.4
Random forest	1,4,7,11,15,20,23,26,30,31,35,36,41,42	98.9	0,2,8,9,12,17,18,19,20,22,26,27	98.8
Decision tree	1,2,4,9,11,12,23,35,36,41	98.2	19,27	98.7
MLP	1,5,6,7,8,9,10,12,14,16,18,19, 20,21,26,31, 35,36,41,42	96.1	1,2,5,6,17,19,22,23,26,27	98.4
K-NN	1,4,7,21,22,29,32,35,38,41,42	97.7	16,19,20,22,27	98.7

Selected row in Fig. 4 illustrates the accuracy of 93.1% with eleven features which five of them are col 1, col 3, col 7, col 21 and col 27.

In Table 2 the selected feature for every algorithm is shown. It should be noted that if the given algorithm represents some groups of features with the same accuracy, a group selected with a smaller number of features and the index feature starts from zero.

Furthermore, the accuracy can be calculated with Eq. (1).

$$\textbf{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+FP+TN+FN}} \tag{1}$$

In Fig. 5 the result in a simple plot to compare datasets accuracy with each other is shown.

5.3. Result

As it is described before, the naïve Bayes, random forest, decision tree, MLP, and K-NN algorithm are implemented. The trained model for 20-times with 5-fold cross-validation is performed and the results for each dataset are collected. After that, the mean of results is taken. The precision, recall, and f-measure of each algorithm, which are calculated with the following equation are shown in Table 1. In front of each algorithm, cells are divided into two cells, one of them for DDoS attack type and the other for detecting the normal type. Each of the criteria is written separately for each data class (Table 3).

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3}$$

$$\text{F} - \text{measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

The given implementation is based on a tool to verify its accuracy. The results are compared with Alkasassbeh et al. [33] results which is shown in Table 4.

In Fig. 6 the difference of accuracy between given implementation and implementation in reference [33] is shown.

Until now the results without considering the determiner in the proxy side is described. Before describing determiner policy, the outputs of all algorithms together will be put in a table with correct output. By a simple look to the result in the first dataset, only 18 records among 17,168 records exist which all of the algorithms have mistaken. There is 2033 records that the algorithms decide different and at least one of them are able to detect right. This result for the second dataset between 713,021 records is 3083 and 9430 respectively.

Determiner can have different policies, for example, here a simple one is considered. It is assumed that if at least two of algorithms determine correct, the whole approach determine correctly otherwise mistake happened. Two datasets with this policy

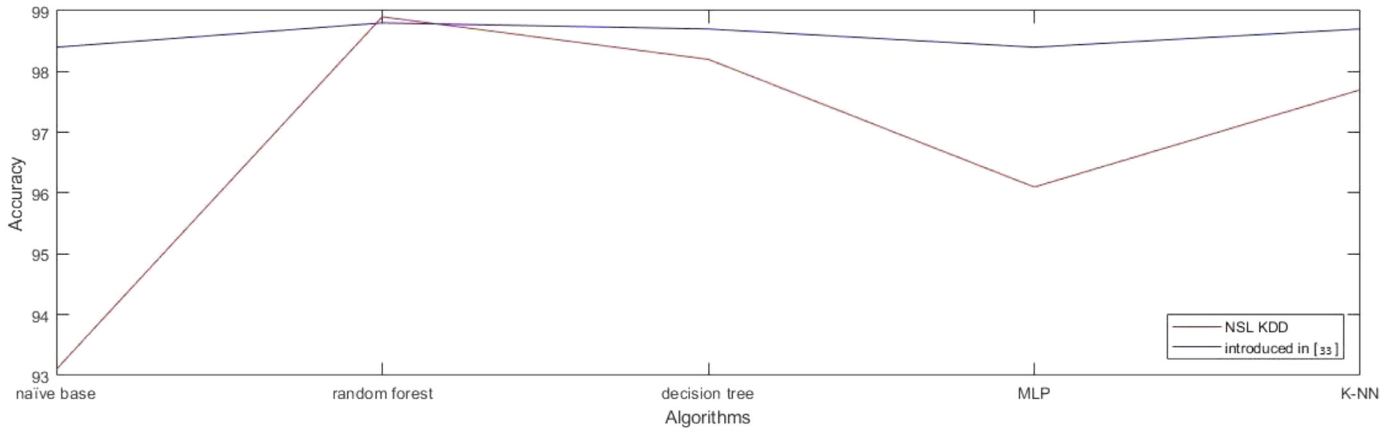


Fig. 5. Accuracy compare between algorithms for two datasets.

Table 3
Evaluation of the result.

Algorithm\Criterion	Type	NSL-KDD			The introduced dataset in [33]		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Naïve Bayes	DDoS	93.6	87.3	92.7	99.9	84.5	91.6
	Normal	94.4	97.5	95	98.2	1	99.1
Random forest	DDoS	99.6	99.8	99.7	99.9	87.3	93.2
	Normal	99.9	99.8	99.8	98.5	1	99.3
Decision tree	DDoS	99.4	99.8	99.6	99.9	87.3	93.1
	Normal	99.9	99.6	99.7	98.5	1	99.3
MLP	DDoS	93.4	91.8	94.9	99.9	86.2	92.5
	Normal	97.5	95.6	96.4	98.4	1	99.2
K-NN	DDoS	99.8	99.8	99.8	99.9	87.3	93.1
	Normal	99.9	99.8	99.9	98.5	1	99.3

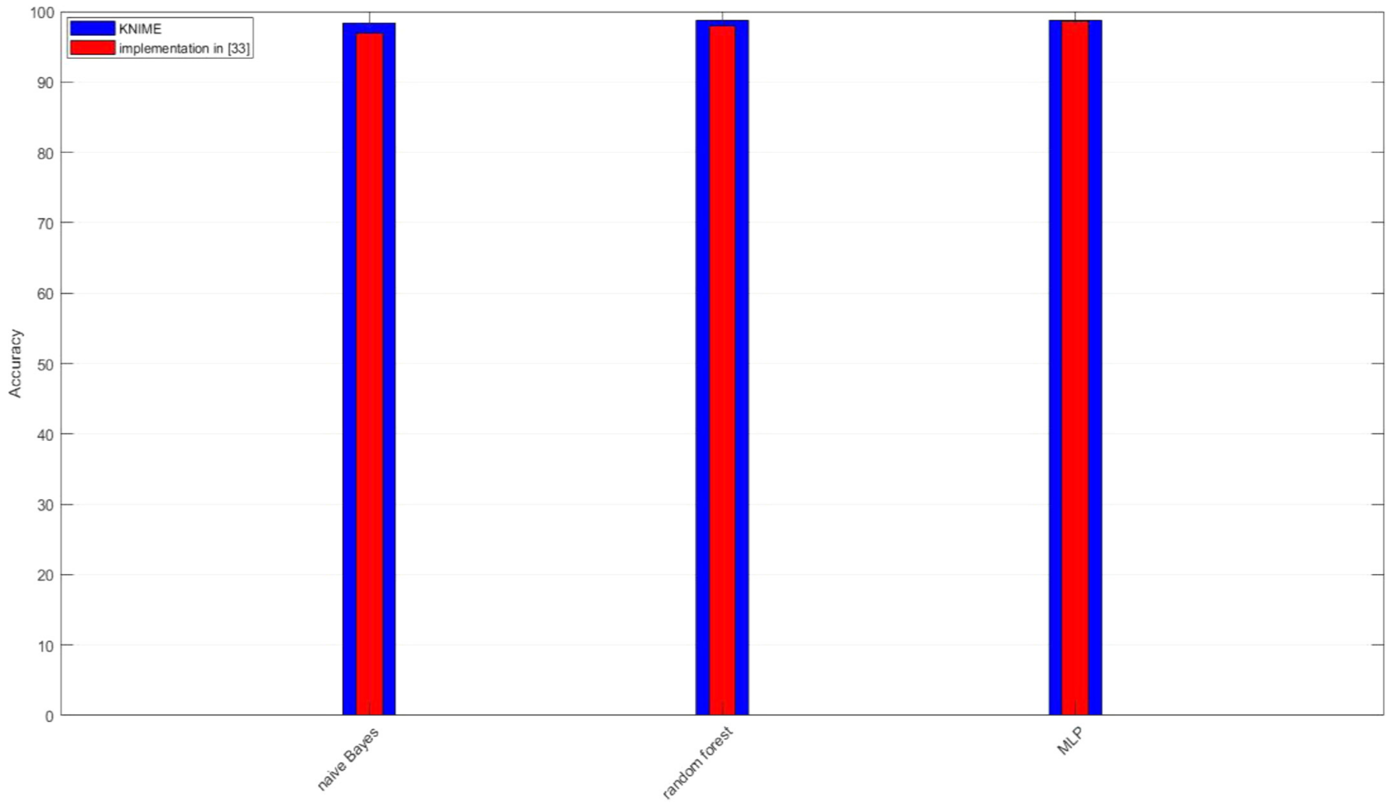


Fig. 6. Accuracy comparing between algorithms for various implementation.

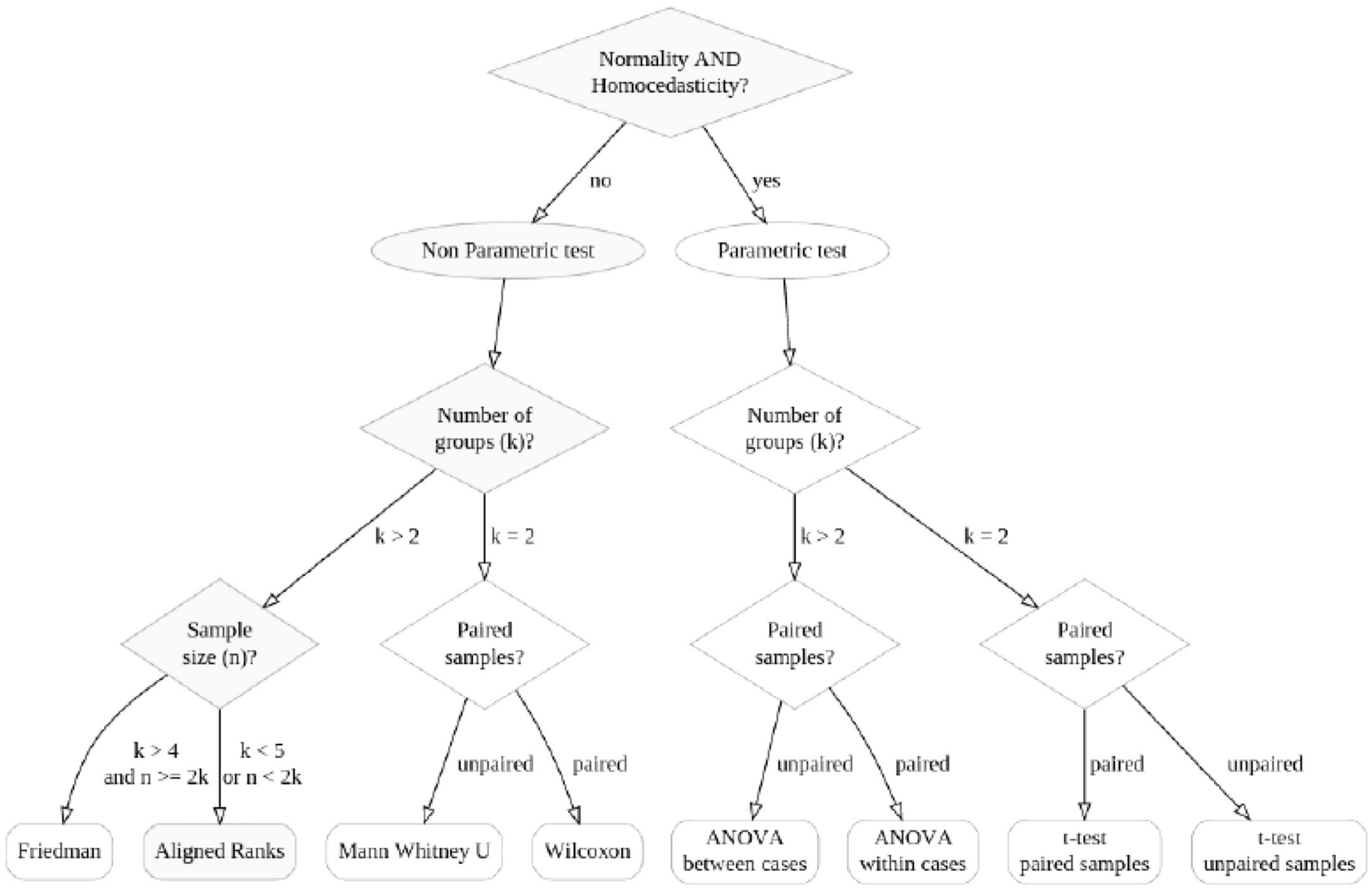


Fig. 7. STAC assistant decision tree [35].

Table 4
Comparison of the accuracy.

Algorithm\Implementation	KNIME	The introduced dataset in [33]
	Accuracy	Accuracy
Naïve Bayes	98.4	96.91
Random forest	98.7	98.02
MLP	98.8	98.63

Table 5
The comparison of True positive rate and false positive rate.

Measure\Dataset	NSL KDD	Introduces dataset in [33]
TPR	0.995	0.873
FPR	0.005	0.00006

is processed. Results are described and calculated true positive rate (TPR) and false positive rate (FPR) in Table 5 according to following equations.

$$\mathrm{TPR} = \frac{TP}{TP+FN} \tag{6}$$

$$\mathrm{FPR} = \frac{FP}{FP+TN} \tag{7}$$

Detecting DDoS attack is considered as the positive class and detecting normal data is considered as the negative class. Therefore:

Variance is a measurement of the spread between numbers from their average value in a data set. Another definition is the average of the squared differences from the mean. For each dataset and algorithm, the variance is calculated separately, for normal and DDoS class. First of all, the normal and DDoS rows in algorithms result are distinguished. For each class, the standard deviation for all columns should be calculated and then squared them and get the average number. In the following, variance formula and results in Table 6 are given. In Table 6 the results are shown according to the variance formula which is shown in Eq. (8).

Table 6
The comparison of data variance.

Algorithm	NSL-KDD		The introduced dataset in [33]	
	Normal	DDoS	Normal	DDoS
Naïve Bayes	0.0362	0.0653	0.0597	0.0201
Random forest	0.0445	0.0479	0.0319	0.0344
Decision tree	0.0377	0.0435	0.0179	0.0569
MLP	0.0206	0.0144	0.0399	0.0257
K-NN	0.0274	0.0584	0.0183	0.0428

$$\sigma^2 = \frac{\sum (X - \mu)^2}{N} \tag{8}$$

Where X is an individual data point, u is mean of data points, and N is total number of data points.

Variance value always will be zero or positive number. A small variance indicates that numbers in the set are near to mean and each other, while a large variance shows that the opposite ones. Small variance is better than a large one and improves generalization for classifiers.

For comparing the examined algorithms, we consider a statistical test using STAC [35]. STAC is a web platform for the comparison of methods with considering statistical tests. If the

Table 7
Aligned-Ranks test (significance level of 0.05).

Statistic	*p*-value	Result
6.35000	0.17449	H0 is accepted

Table 8
Ranking of all algorithms.

Algorithms	Ranking
Random forest	8.50
Decision tree	7.25
K-NN	6.75
MLP	2.75
Naïve Bayes	2.25

Table 9
Aligned ranks test (significance level of 0.05).

Comparison	Statistic	Adjusted *p*-value	Result
Naïve Bayes vs Random forest	2.06431	0.38989	H0 is accepted
MLP vs Random forest	1.89916	0.51789	H0 is accepted
Decision tree vs Naïve Bayes	1.65145	0.78918	H0 is accepted
MLP vs Decision tree	1.48630	0.96040	H0 is accepted
K-NN vs naïve Bayes	1.48630	0.96040	H0 is accepted
MLP vs K-NN	1.32116	0.96040	H0 is accepted
K-NN vs Random forest	0.57801	1.00000	H0 is accepted
Decision tree vs Random forest	0.41286	1.00000	H0 is accepted
MLP vs Naïve Bayes	0.16514	1.00000	H0 is accepted
Decision tree vs K-NN	0.16514	1.00000	H0 is accepted

data have a distribution (usually normal), then a parametric test can be used; otherwise, if the data distribution are not specified, we use a nonparametric test. Therefore, we have a non-parametric test, the number of algorithms are 5 and the number of datasets is 2 ($k = 5$ and $n = 2$) thus based on STAC assistant decision tree, Aligned-Ranks test will be chosen. Fig. 7 shows path to how Aligned-Ranks test choose.

The goal of the statistical test is to compare the accuracy of the algorithms thus we compare the algorithms two by two.

First of all, we normalize accuracies in Table 4 between [0, 1] and give the values to STAC as input. We apply Aligned-Ranks test and set the post-hoc parameter to holm and significant level to 0.05 which is shown in Table 7. H0 is accepted when the p-value is greater than the significant level.

Table 8 shows the rank of algorithms in STAC. As shown in Table 8, naïve Bayes gets the minimum score and random forest gets the maximum score.

Another output is obtained to compare algorithms two by two, which is shows in Table 9.

6. Conclusions

In this paper, we proposed the hybrid framework to detect the DDoS attack. Based on the results, the processes into two sides divided. Each side performed on its work therefore the speed in order to organize work with this trick is improved. KNIME presented its performance when is compared to other works hence the given results are reliable. Random forest in both datasets analyzing reappearance better results but in a special situation any of other algorithms may work better. This is one of the reasons to use the number of classifications. As it can be seen each of the algorithms selects a different subset of features thus each one performs based on specified features of attack. Accordingly, a wide range of attacks based on the behavior is detected. Using the number of classification algorithm instead of one in a long time gives the ability to face an unknown attack better than the previous method. Each algorithm included its weaknesses and strengths thus is used the combination of algorithms to take a better system detection. Another property of this paper is store profile in a database of the attack to prevent over-process on stream data, and is able to compare data features to the provided profile database with the a-divergence test.

Conflict of interest

In the future, we will extend our framework to detect attack by considering unsupervised learning methods.

References

[1] K.N. Mallikarjunan, K. Muthupriya, S.M. Shalinie, A survey of distributed denial of service attack, in: Intelligent Systems and Control (ISCO), 2016 10th International Conference on, IEEE, 2016, pp. 1–6.
[2] P.J. Criscuolo, Distributed Denial of Service: Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht, CIAC-2319, Univ Livermore Radiation Lab, California, 2000.
[3] B. Nagpal, P. Sharma, N. Chauhan, A. Panesar, DDoS tools: classification, analysis and comparison, in: Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on, IEEE, 2015, pp. 342–346.
[4] M.I. Jordan, T.M. Mitchell, Machine learning: trends, perspectives, and prospects, Science 349 (6245) (2015) 255–260.
[5] I.H. Witten, E. Frank, M.A. Hall, C.J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2016.
[6] A.L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Commun. Surv. Tutor. 18 (2) (2016) 1153–1176.
[7] R.M. Berthold, N. Cebron, F. Dill, T.R. Gabriel, T. Kotter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, B. Wiswedel, Studies in Classification, Data Analysis, and Knowledge Organization, Springer, 2007 knime: the konstanz information miner.
[8] J. Qiu, Q. Wu, G. Ding, Y. Xu, S. Feng, A survey of machine learning for big data processing, EURASIP J. Adv. Signal Process. 2016 (1) (2016) 67.
[9] B. Singh, S. Panda, Defending against DDOS flooding attacks-a data streaming approach, Int. J. Comput. IT 1 (2015) 38–44.
[10] S.T. Zargar, J. Joshi, D. Tipper, A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks, IEEE Commun. Surv. Tutor. 15 (4) (2013) 2046–2069.
[11] K.M. Prasad, A.R.M. Reddy, K.V. Rao, DoS and DDoS attacks: defense, detection and traceback mechanisms-a survey, Glob. J. Comput. Sci. Technol. 14 (2014) 15–32.
[12] J. Mirkovic, G. Prier, P. Reiher, Attacking DDoS at the source, in: Network Protocols, 2002. Proceedings. 10th IEEE International Conference on, IEEE, 2002, pp. 312–321.
[13] S. Behal, K. Kumar, M. Sachdeva, D-FACE: an anomaly based distributed approach for early detection of DDoS attacks and flash events, J. Netw. Comput. Appl. 111 (2018) 49–63.
[14] M. Chen, S. Mao, Y. Liu, Big data: a survey, Mob. Netw. Appl. 19 (2) (2014) 171–209.
[15] L. Wang, R. Jones, Big data analytics for network intrusion detection: a survey, Int. J. Netw. Commun. 7 (1) (2017) 24–31.
[16] V. Lemaire, C. Salperwyck, A. Bondu, A survey on supervised classification on data streams, in: European Business Intelligence Summer School, Springer, 2014, pp. 88–125.
[17] D. Namiot, On big data stream processing, Int. J. Open Inf. Technol. 3 (8) (2015).
[18] V. Sekar, N.G. Duffield, O. Spatscheck, J.E. van der Merwe, H. Zhang, LADS: large-scale automated DDoS detection system, in: USENIX Annual Technical Conference, General Track, 2006, pp. 171–184.
[19] H. Rahmani, N. Sahli, F. Kammoun, Joint entropy analysis model for DDoS attack detection, in: Information Assurance and Security, 2009. IAS'09. Fifth International Conference on, 2, IEEE, 2009, pp. 267–271.
[20] B.B. Gupta, R.C. Joshi, M. Misra, ANN based scheme to predict number of zombies in a DDoS attack, IJ Netw. Secur. 14 (2) (2012) 61–70.
[21] J. François, I. Aib, R. Boutaba, FireCol: a collaborative protection network for the detection of flooding DDoS attacks, IEEE/ACM Trans. Netw. (TON) 20 (6) (2012) 1828–1841.
[22] M. Barati, A. Abdullah, N.I. Udzir, R. Mahmod, N. Mustapha, Distributed Denial of Service detection using hybrid machine learning technique, in: Biometrics and Security Technologies (ISBAST), 2014 International Symposium on, IEEE, 2014, pp. 268–273.
[23] A. Fadlil, I. Riadi, S. Aji, A novel DDoS attack detection based on Gaussian Naive Bayes, Bull. Electr. Eng. Inform. 6 (2) (2017) 140–148.
[24] D. Kim, K.Y. Lee, Detection of DDoS attack on the client side using support vector machine, Int. J. Appl. Eng. Res. 12 (20) (2017) 9909–9913.
[25] M.A.M. Yusof, F.H.M. Ali, M.Y. Darus, Detection and defense algorithms of different types of DDoS attacks using machine learning, in: International Conference on Computational Science and Technology, Springer, 2017, pp. 370–379.

[26] B. Zhou, J. Li, J. Wu, S. Guo, Y. Gu, Z. Li, Machine-learning-based online distributed denial-of-service attack detection using spark streaming, in: in *2018 IEEE International Conference on Communications (ICC)*, IEEE, 2018, pp. 1–6.

[27] M. Idhammad, K. Afdel, M. Belouch, Semi-supervised machine learning approach for DDoS detection, Appl. Intell. 48. (2018) 3193–3208.

[28] K.N. Mallikarjunan, A. Bhuvaneshwaran, K. Sundarakantham, S.M. Shalinie, DDAM: detecting DDoS attacks using machine learning approach, in: Computational Intelligence: Theories, Applications and Future Directions-Volume I, Springer, 2019, pp. 261–273.

[29] P. Xiao, W. Qu, H. Qi, Z.J.C.C. Li, Detecting DDoS attacks against data center with correlation analysis, Comput. Commun., 67 (2015) 66–74.

[30] I. Cano and M.R. Khan, "ASML: aAutomatic streaming machine learning," ed.

[31] "Nsl-kdd data set for network-based intrusion detection systems." Available on: http://nsl.cs.unb.ca/KDD/NSL-KDD.html, March 2009.

[32] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, in: Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on, IEEE, 2009, pp. 1–6.

[33] M. Alkasassbeh, G. Al-Naymat, A. Hassanat, M. Almseidin, Detecting distributed denial of service attacks using data mining techniques, Int. J. Adv. Comput. Sci. Appl. 7 (1) (2016).

[34] M. Mehta, J. Rissanen, R. Agrawal, MDL-based decision tree pruning, in: KDD, 21, 1995, pp. 216–221.

[35] I. Rodríguez-Fdez, A. Canosa, M. Mucientes, A. Bugarín, STAC: a web platform for the comparison of algorithms using statistical tests, in: Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on, IEEE, 2015, pp. 1–8.

Soodeh Hosseini received B.S. degree in computer science (2004) from Shahid Bahonar University of Kerman and the M.Sc. and Ph.D. in computer engineering (software from Iran University of Science and Technology, in 2007 and 2016, respectively. Her main research interests include machine learning, cyber security and computer simulation. She has published several papers in international journals and conferences. Currently, she is an assistant professor at Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran.

Mehrdad Azizi received B.S. degree in computer science (2016) from Vali-e-Asr University of Rafsanjan and now, his pursuing M.S degree in Shahid Bahonar University of Kerman. His area of interests include machine learning, cyber security, deep learning and computer vision.