UDC 004.056.5
doi:10.31799/1684-8853-2020-3-59-70

Comparison of machine learning algorithms for DDoS attack detection in SDN

D. T. Le[a]***,*** *PhD, Lecturer, orcid.org/0000-0003-3735-0314, letranduc@dut.udn.vn*
M. H. Dao[a]***,*** *Master Student, orcid.org/0000-0002-0998-6078*
Q. L. T. Nguyen[a]***,*** *M. Sc., Lecturer, orcid.org/0000-0003-4578-9925*
[a]*The University of Danang — University of Science and Technology, Information Technology Faculty, 54 Nguyen Luong Bang, 550000, Da Nang, Vietnam*

Introduction: *Distributed denial-of-service (DDoS) has become a common attack type in cyber security. Apart from the conventional DDoS attacks, software-defined networks also face some other typical DDoS attacks, such as flow-table attack or controller attack. One of the most recent solutions to detect a DDoS attack is using machine learning algorithms to classify the traffic.* ***Purpose:*** *Analysis of applying machine learning algorithms in order to prevent DDoS attacks in software-defined network.* ***Results:*** *A comparison of six algorithms (random forest, decision tree, naive Bayes, support vector machine, multilayer perceptron, k-nearest neighbors) with accuracy and process time as the criteria has shown that a decision tree and naïve Bayes are the most suitable algorithms for DDoS attack detection. As compared to other algorithms, they have higher accuracy, faster processing time and lower resource consumption. The main features that identify malicious traffic compared to normal one are the number of bytes in a flow, time flow, Ethernet source address, and Ethernet destination address. A flow-table attack can be detected easier than a bandwidth attack, as all the six algorithms can predict this type with a high accuracy.* ***Practical relevance:*** *Important features which play a supporting role in correct data classification facilitate the development of a DDoS protection system with a smaller dataset, focusing only on the necessary data. The algorithms more suitable for machine learning can help us to detect DDoS attacks in software-defined networks more accurately.*

Keywords *— DDoS, machine learning algorithms, flow-table attack, bandwidth attack.*

For citation: Le D. T., Dao M. H., Nguyen Q. L. T. Comparison of machine learning algorithms for DDoS attack detection in SDN. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2020, no. 3, pp. 59–70. doi:10.31799/1684-8853-2020-3-59-70

Introduction

Nowadays, software defined network (SDN) is becoming increasingly popular due to the benefits it provides, such as scalability, flexibility, monitoring [1]. SDN architecture separates the network control from forwarding devices and enables the controller to become directly programmable. The controller processes the packets and decides whether the packets will be forwarded in the switch or dropped. Due to its centralized nature, the controller can get a global view of the network, and it helps the network administrators to adjust the network traffic flow dynamically [2]. Besides, for network components to interact with each other, several application programming interfaces were developed with this network model, typically the OpenFlow (OF) protocol [3].

However, the SDN network also faces many security threats [4]. When management becomes centralized, it will be easier for the administration, but it will also be easier to be collapsed under attacks. One of the attacks that have the most devastating effect on an SDN network is the distributed denial-of-service (DDoS) attack [5]. It is explained based on the distinct characteristics of the SDN network. In the SDN network, besides conventional DDoS attacks by taking up network resources, causing the system to be paralyzed, we also face other types of attacks. For example, instead of attacking with a large number of large packets to occupy bandwidth, the attacker will constantly flood the network with strange packets so that the controller is forced to create new rules for these packets and write them in flow-table. Then, the table on the switch will increase until there is no more space to new rules, and as a consequence, the time to respond to each new requisition increased [6].

There have been several proposed solutions to solve this problem. For example, drop packets, block port, redirection, control bandwidth, deep packet inspection, network reconfiguration, and topology change; each solution has its advantages and disadvantages [5]. However, for the above attack mitigation techniques to be effective, SDN needs to implement effective DDoS attack detection techniques. The paper [5] introduces several methods for detecting DDoS attacks, such as using entropy [7], traffic pattern analysis [8], connection rate [9], machine learning [6, 10]. Among them, DDoS detection techniques using machine learning have received much attention in the computational intelligence community [11].

This technique is not new. There have been many studies considering the ability of machine learning to classify traffic on the SDN environment as in

[12–15]. However, due to the variety of algorithms, each machine learning algorithm (simplified here by ML-algorithms) has its own approach to the problem, maybe appropriate, maybe not, but it gives us more options to solve the problem, as well as to pick out the algorithm that best suits the goal of detecting DDoS attack.

In this paper, besides focusing on how to apply machine learning to detect DDoS in the SDN environment, we will implement six different ML-algorithms, making comparisons based on some criteria to expand the choice and finding the optimal solution. These algorithms are random forest (RF), decision tree (DT), naive Bayes (NB), support vector machine (SVM), multilayer perceptron (MLP), k-nearest neighbors (KNN), all supported by Python libraries.

Related works

In [16], Braga et al. proposed a lightweight method for DDoS attack detection based on traffic flow features. This method is implemented over a NOX-based network, where OF switches keep Flow Tables with statistics about all active flows. This system monitors NOX switches at regular intervals and uses self-organizing maps to classify the traffic as normal or malicious.

The authors in [17] introduced a deep learning based multi-vector DDoS detection system in a SDN environment. A DDoS detection system that incorporates stacked autoencoder based deep learning approach in an SDN environment was implemented. The authors evaluated its performance on a dataset that consists of normal Internet traffic and various DDoS attacks. However, as every packet has to be collected for extracting features, this approach may limit the performance of the controller in large networks.

In [18], Giotis et al. combined an OpenFlow and sFlow for anomaly detection to reduce processing overhead in native OF statistics collection. It designs a modular mechanism that permits anomaly detection and mitigation on SDN environments, including collector, anomaly detection and anomaly mitigation. It leverages the packet sampling capability of sFlow to acquire scalability improvements and to reduce the required communication between switches and OF controllers. However, as the implementation was based on flow sampling using sFlow, false-positive was quite high in attack detection.

In [19], Ashraf et al. aimed to handle intrusion and DDoS attacks in the SDN environment applying machine learning techniques. However, they only analyzed various machine learning techniques, such as support vector machine, fuzzy logic, decision tree, neural networks, and Bayesian networks (BayesNet), which can be used to detect DDoS attacks in the networking system and no further explanation of how to detect and mitigate DDoS attack was given.

Kokila et al. in [13] explored the possibility of launching DDoS attacks and detection of DDoS using the SVM classifier. The experiments are carried out using the DARPA dataset. They suggested that the use of a support vector machine for detection of DDoS with a previously trained dataset will give the least false-positive results compared with other machine learning techniques.

Dao et al. in [20] presented a solution based on the IP-filtering technique to defeat DDoS attacks. The proposed scheme analyzes user behaviour and uses it to assign the timeouts for the flow entries. Long timeouts are used for trusted users' flows, while a short timeout is assigned for malicious ones. It works well when the attack traffic is not very massive. However, this solution drops all malicious traffic, which may be problematic for false-positive flows.

In [21], Nanda et al. propose using machine learning algorithms, trained on historical network attack data, to identify the potential malicious connections and potential attack destinations. They used four ML-algorithms: DT, BayesNet, decision table and NB to predict the host that will be attacked based on the historical data. The SDN controller uses the prediction results to define security rules to protect the potentially vulnerable hosts and restrict the access of potential attackers by blocking the entire subnet.

Our paper is motivated by Santos's paper [6], in which the authors managed to exploit different kinds of machine learning algorithms to avoid three types of DDoS attacks (controller attack, flow table attack, and bandwidth attack). However, they only focused on the typical attack type of SDN networks. In our paper, we are going to consider both conventional and typical DDoS attacks. We also add more ML-algorithms as well as modify some parameters to make a comprehensive comparison and try to find out appropriate algorithms for detecting DDoS attacks in the SDN environment.

Machine learning algorithms for DDoS detection

In this paper, we will implement six different ML-algorithms, making comparisons based on some criteria to expand the choice and finding the optimal solution. These algorithms are RF, DT, NB, SVM, MLP, KNN [22].

Decision tree. The DT is one of the classification techniques, which performs classification through a learning tree. In the tree, each node represents a feature (attribute) of a data, all branches represent the conjunctions of features that lead to classifications, and each leaf node is a class label. The unlabeled sample can be classified by comparing its feature values with the nodes of the DT. The DT has many advantages, such as intuitive knowledge expression, simple implementation, and high classification accuracy. However, due to its instability, even a small change in the training dataset can result in significant changes in the DT-model.

Random forest. The RF-algorithm, also known as random decision forest, can be used for classification and regression tasks. A RF consists of many DTs. This algorithm works well on the large training dataset and reduces instability (relative to DT). However, it has low training speed. The steps to classify a new data sample by using a RF-algorithm are: a) put the data sample to each tree in the forest; b) each tree gives a classification result, which is the tree's "vote"; c) the data sample will be classified into the class, which has the most votes.

k-nearest neighbors. The KNN is a supervised learning technique, where the classification of a data sample is determined based on the k nearest neighbors of that unclassified sample. The process of the KNN-algorithm is very simple: if most of the KNN belong to a specific class, the unclassified sample will be classified into that class. This algorithm is simple to implement but computationally expensive due to the distance calculation of each training data sample to classify a new sample.

Naïve Bayes uses Bayesian theory that predicts the type of the unknown samples based on

■ ***Table 1*. Hyperparameters and ML-algorithms**

Models	Hyperparameters	Description
Decision tree	criterion	The function to measure the quality of a split
	splitter	The strategy used to choose the split at each node
	min_samples_split	The minimum number of samples required to split an internal node
	min_samples_leaf	The minimum number of samples required to be at a leaf node
Random forest	n_estimators	The number of trees in the forest
	criterion	The function to measure the quality of a split
	min_samples_split	The minimum number of samples required to split an internal node
	min_samples_leaf	The minimum number of samples required to be at a leaf node
Naive Bayes	var_smoothing	A portion of the largest variance of all features that is added to variances for calculation stability
k-nearest neighbor	n_neighbors	The number of neighbors to use
	weights	Weight function used in prediction
	leaf_size	Leaf size passed to BallTree or KDTree
	p	Power parameter for the Minkowski metric
	metric	The distance metric to use for the tree
	algorithm	Auto between: ball_tree, kd_tree, brute
Support vector machine	Kernel	Specifies the kernel type to be used in the algorithm
	Gamma	Kernel coefficient for 'rbf', 'poly' and 'sigmoid'
	C	Regularization parameter
	Tol	Tolerance for stopping criterion
	max_iter	Hard limit on iterations within solver
Multilayer perceptron	hidden_layer_sizes	The ith element represents the number of neurons in the ith hidden layer
	activation	Activation function for the hidden layer
	solver	The solver for weight optimization
	alpha	L2 penalty (regularization term) parameter
	max_iter	The maximum number of iterations
	tol	Tolerance for optimization
	max_fun	The maximum number of loss function calls

prior probability using the training samples. The Bayesian classification model relies on statistical analysis and Bayesian theory that consists of the Bayesian learning. The NB-algorithm operates by segregating the training set into an attribute vector and a decision variable. The algorithm also assumes that every member of the attribute vector independently acts on the decision variables.

Support vector machine. SVM is another popular supervised learning method, which has been widely used in classification and pattern recognition. The basic idea of SVM is to map the input vectors into a high-dimensional feature space. This mapping is achieved by applying different kernel functions, such as linear, polynomial and radial based function (RBF). The objective of SVM is to find a separating hyperplane in the feature space to maximize the margin between different classes. The disadvantage of this algorithm is hard to train large datasets because the training is computationally expensive.

Multilayer perceptron. The MLP is a class of feedforward artificial neural network and has been widely adopted neural network for intrusion detection in conventional systems. An MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

Each algorithm has its own strengths. We will test each algorithm as well as compare them together to select the optimal algorithm for the detection of DDoS attacks in the SDN network. Table 1 shows the hyperparameters used in this experiment associated with the respective machine learning algorithm.

The studied features used to build the model for the algorithms are shown in Table 2.

The number of studied features is up to 14, which is almost all of the data that we can get from the flow-table through OpenFlow Switch. Among the above features, not all features help to detect abnormal and normal traffic classification. However, at this time, it is not known which features will play a decisive role in detecting DDoS attacks, so the models will be built based on all these features. At the end of the experiment, we can review and take out the important features table to find out which features will play a decisive role in this case. Because the characteristics of each algorithm are different, important features are particular parameters that can only be derived when studying DT and RF. However, these features still help identify the features needed to reduce data in building models of other algorithms.

■ *Table 2.* **Description of studied features**

№	Features	Description
0	Byte_count	Number of bytes in a flow
1	Cookie	Opaque controller-issued identifier
2	Eth_src	Ethernet source address
3	Eth_dst	Ethernet destination address
4	Duration_nsec	Time flow has been alive in nanoseconds
5	Duration_sec	Time flow has been alive in seconds
6	Hard_timeout	Max time before discarding (seconds)
7	Idle_timeout	Idle time before discarding (seconds)
8	In_port	Port ID
9	Max_len	Max length to send to the controller
10	Packet_count	Number of packets in the flows
11	Priority	The priority level of a flow entry
12	Port	Output port
13	Table_id	The ID of the table to put the flow in
14	Type	Type of action

Goals and implementation plan

Goals

The primary purpose of our experiment is to find ways to apply machine learning to detect DDoS attacks in SDN networks. Besides, another goal that we are aiming at in this paper is to compare different ML-algorithms as a solution to the problem, because each algorithm has its own characteristics.

Based on [23], the basic criteria to evaluate a model in detecting abnormal traffic such as DDoS include accuracy, data quality, correctness, and efficiency. In this paper, since all the tests take place in a simulation environment, and there is always a difference between simulation data and actual collected data, we will not perform an evaluation based on data quality.

Usually, with classification problems for any model of machine learning, accuracy criterion is a suitable criterion for evaluation. It indicates how much percentage of a model's accuracy is rated, which makes it easy to visualize. For processing time, this is the time for a model to classify a flow from input into normal traffic or abnormal traffic. In other words, the accuracy and processing time criteria will represent the efficacy and efficiency of the machine learning model, respectively. Also, to evaluate the algorithms more objectively, we rely on information from the receiver operating characteristic (ROC) curve (correctness) [6] to be able to choose the suitable model.

Planning

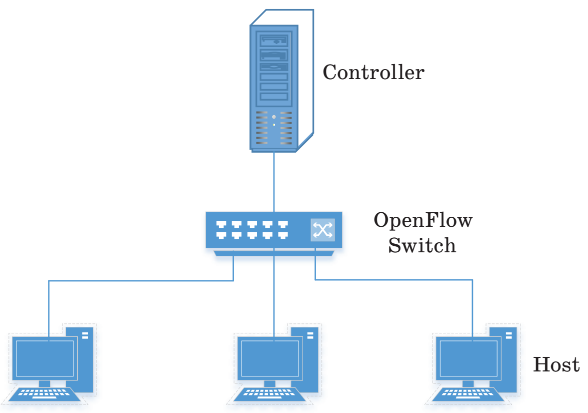
Network architecture

The entire experiment is carried out on Ubuntu 18.04 virtual machine VMware with hardware configuration Core i5-5200U (2.2 GHz, 4 cores, 4 processors), 2 GB of RAM, and 40 GB hard drive space. Mininet VN (version 2.3.0d6) is used for creating the SDN network with an RYU controller (version 4.32), one OpenFlow Switch and three hosts, as shown in Fig. 1.

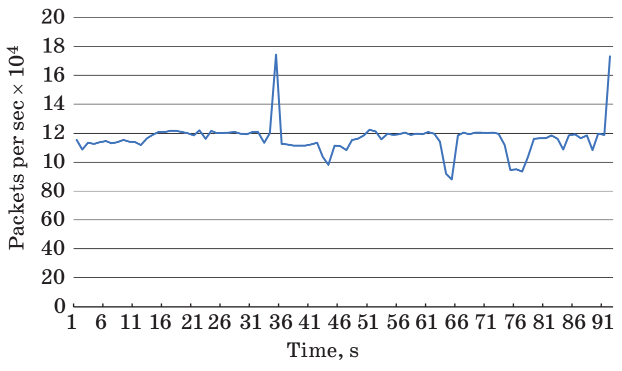
Methods of attacks

In addition to conventional attacks to a host or a group of hosts such as UDP flood, ping flood or smurf attack (collectively called bandwidth attacks), SDN network can also be attacked by new DDoS attack types due to its own structural characteristics such as controller attack, flow-table attack [6]. Therefore, to be able to study objectively and more fully, we will try the bandwidth attack and the flow-table attack in SDN at the same time.

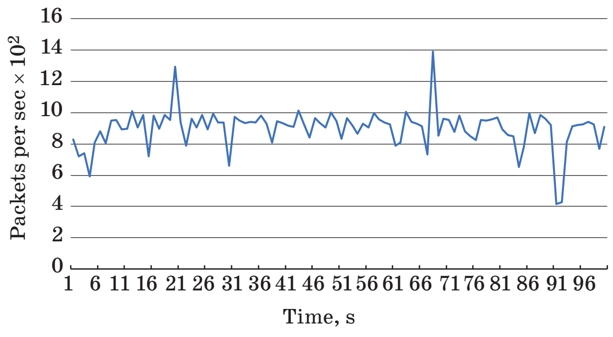
For flow-table attack, we will use Scapy tool to continuously send packets from different addresses (there are 20,000 randomly generated Ips saved in a file) to the attacked device. When the switch receives this type of packet, it creates a new rule and adds to its flow-table. As the number of incoming packets grows, the number of entries in flow-table increases and leads to overloading, causing a delay in responding to other requests from the controller.



■ *Fig. 1.* SDN network architecture for the experiment



■ *Fig. 2.* Bandwidth flooding traffic



■ *Fig. 3.* Normal traffic

For bandwidth attack, we will focus on taking up the network bandwidth by continuously flooding large packets (256–512 KB) to hosts in the network. We will combine many different types, including ICMP flood, TCP SYN flood and UDP flood. We use the hping3 tool to get the best results. Unlike flow-table attack when it only focuses on increasing the number of entries in the flow-table as quickly as possible, for this attack method instead of focusing on the number of attacker machines, we focus on packet-flood rate (pps — packets per second) and packet size to take up the network bandwidth.

Figure 2 shows the average network traffic under a DDoS attack with bandwidth attack (pps ~ 120,000).

We can see a huge difference comparing to normal traffic shown in Fig. 3. Normal traffic is simulated using Scapy. UDP, HTTP, ICMP packets are continuously created for sending inside the network, generating reasonable traffic (pps ~ 1.000).

Experiment Execution

Data Collection

To collect the necessary data from the switch's flow-table, we built a separate module. This module is responsible for reading entries in the flow-table every second, recording the required information into a data file, and labelling it.

We then use hping3 from one of the three hosts inside the network (see Fig. 1) to create a bandwidth attack on the remaining hosts. Similarly, we use Scapy to create flow-table attack as well as normal traffic.

In the end, the data collected is 7500 data for each type (2500 data of the dataflow table for bandwidth attack, 2500 for flow-table attack, and 2500 for normal traffic). Based on this data set, we will create two separate datasets: train dataset and test dataset for the next process.

Building model

We use the training dataset prepared above to build the models. After that, we will check with the test dataset to get the best results. To avoid model overfitting (especially for NB, KNN, SVM, and MLP), standardization of the values of the features was applied to the data using *StandardScaler* in scikit-learn [24].

Next is the process of tuning hyperparameters. We use the *GridSearch* technique (from Sklearn Library) to find the best hyperparameters set. It helps to build a suitable model that is highly effective.

After having obtained a reasonable hyperparameter set, during the next training period, we use cross-validation to avoid algorithms' overfitting with the training dataset. Specifically, we use *StratifiedKFold* [24] with ten folds and then evaluate the returned results, from which the conclusion is made.

Results

After the process of tuning hyperparameter, we obtain the following parameters, as in Table 3.

■ ***Table 3***. The best hyperparameters set for each ML-algorithm

Model	Hyperparameter	Value
Random forest	n_estimators	10
	criterion	Gini
	min_samples_split	2
	min_samples_leaf	1
Decision tree	Criterion	Gini
	Splitter	Best
	min_samples_split	2
	min_samples_leaf	1
Naive Bayes	var_smoothing	1e-9
k-nearest neighbors	n_neighbors	3
	weights	Uniform
	leaf_size	30
	p	2
	metric	Minkowski
	algorithm	Auto
Support vector machine	Kernel	Rbf
	Gamma	Auto
	C	1e+5
	Tol	1e-3
	Max_iter	-1
Multilayer perceptron	hidden_layer_sizes	(5,)
	activation	Relu
	solver	Lbfgs
	alpha	1e-3
	max_iter	2000
	Tol	1e-4
	max_fun	2000

The main objective of this study is to apply machine learning to detect DDoS attacks, compare algorithms, and build a model that can classify as many types of traffic as possible. Therefore, from the initial data (7500 data of dataflow table, 2500 of each type), we will create six different datasets, including train/test datasets for normal traffic and bandwidth attack traffic (ratio 1:1); train/test datasets for normal traffic and flow-table attack traffic (ratio 1:1); finally, train/test datasets for all three traffic types at once (ratio 1:1:1).

To evaluate the accuracy of the ML-algorithms for each attack simulated, we use the following formula [6]:

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Total of samples}}.$$

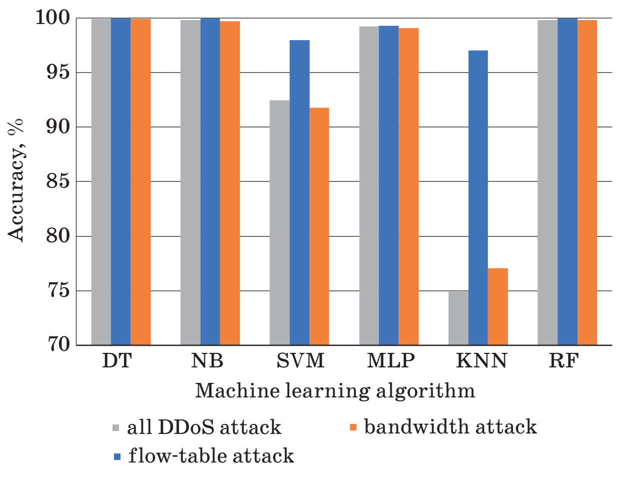
The accuracy is a statistical value that determines how close our ML-algorithm is to the ideal. If this value is 1 (100 %), it means that the algorithm has no error and classifies the data perfectly.

Figure 4 shows a comparison of the accuracy of ML-algorithms.

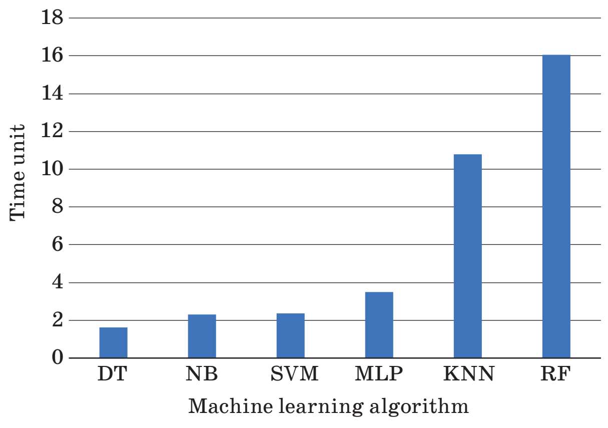
It is easy to realize that for the current problem, the lazy learner algorithm — KNN is entirely inappropriate. It gives results with low accuracy for bandwidth attack. For flow-table attack, although the results are quite good, it is inferior to the remaining algorithms.

The two algorithms SVM and MLP are useful algorithms for this problem with high accuracy. However, SVM still has many errors in the classification of bandwidth attack.

The other three algorithms are RF, DT, and NB are excellent algorithms with an almost absolute precise classification capability.



■ ***Fig. 4.*** Accuracies of ML-algorithms for each DDoS attack



■ ***Fig. 5.*** Relative time to process

Besides accuracy, we also use process time for comparison. Fig. 5 presents this comparison.

As we can see, the two algorithms RF and KNN, take too much time to process compared to other algorithms. Among the remaining four algorithms, although the DT has a faster processing speed than NB, SVM, and MLP, all four algorithms show that they are consistent with the traffic classification.

Besides, in this paper, we also build ROC curve graphs in the model evaluation phase by cross-validation for each algorithm in each attack.

The ROC curve represents a relation between true positive rate — represented by the percentage of data classified as malicious that is really malicious and false positive rate — the percentage of data classified as normal, but that is malicious. This curve is very used in the machine learning to choose a good point for the classifiers, given by the point above the central curve in which the distance between them is maximum.

To determine a good model, we need to consider the shape of the curve as well as the rate of false prediction and the rate of omission depending on the characteristics of each specific case.

In the graphs, we also show the area under curve (AUC) metric, that is, the area under the curve. When this metric is higher, then the classification is better. Fig. 6, *a–f* presents the ROC curve for all algorithms.

Analysis and interpretation

Based on the results we have obtained above, we can say that the type of algorithm lazy learner — KNN is not appropriate for DDoS detection. Because the accuracy prediction rate is low, and it takes a lot of processing time. The reason may be due to the similarity between the traffics at the start of the attack. It leads to misjudging the results because the evaluation is based on nearby neighbors. At the same time, for KNN, the process to make predictions always takes place when new data is received, meaning it requires a longer time to calculate and produce results.

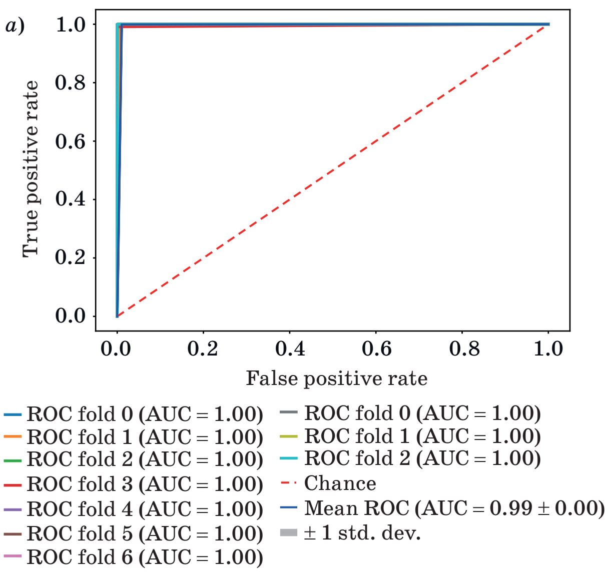
The two algorithms, SVM and MLP, are useful algorithms, capable of applying in detecting DDoS attack with high accuracy and the processing speed is not slow, which is at an average level. However, looking at ROC curve with cross-validation, we found that there are still quite large errors for some data groups; this is relatively understandable because, for SVM, when the noise appears, the hyperplane cannot divide the data exactly, but most are still acceptable.

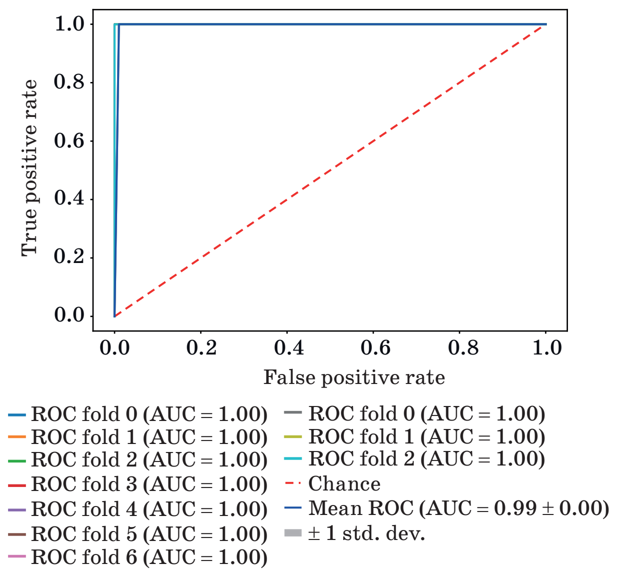
The other three algorithms, RF, DT, and NB, are all excellent algorithms with almost exact classification ability, showing the suitability of these algorithms for the classification of anomalies traffic and normal traffic. Nevertheless, although the RF has almost absolute accuracy, it takes a lot of time to process. The number of trees (10 trees) explains this. Each tree consumes a particular time, resulting in significantly increased time. Therefore, when we need high processing speed and low resource consumption, RF will not be appreciated as DT and NB algorithms.

In contrast to RF, both DT and NB algorithms have very fast processing speed. For the DT-algorithm, the fast processing speed is explained by its advantage. After training to build a decision, the next classification of this algorithm will not need much calculation. For NB, this is always considered an easy algorithm to implement and train even with small data sets. NB is a lightweight algorithm, but the results are still very good.

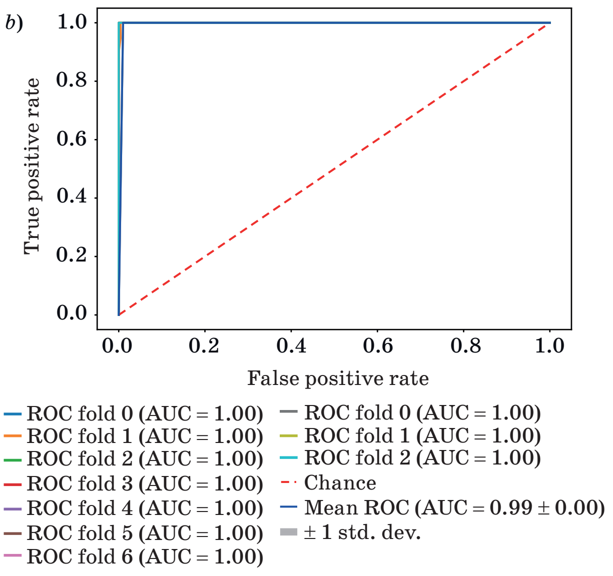
With such results, DT and NB are suitable algorithms for the problem of detecting DDoS attacks in the SDN network.

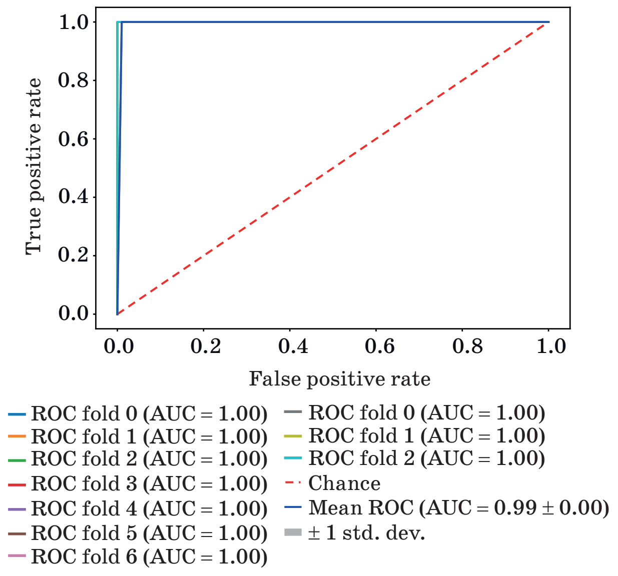
a)



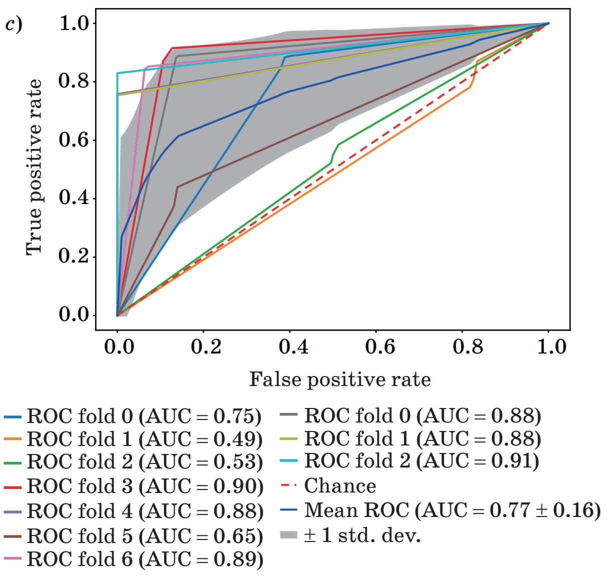


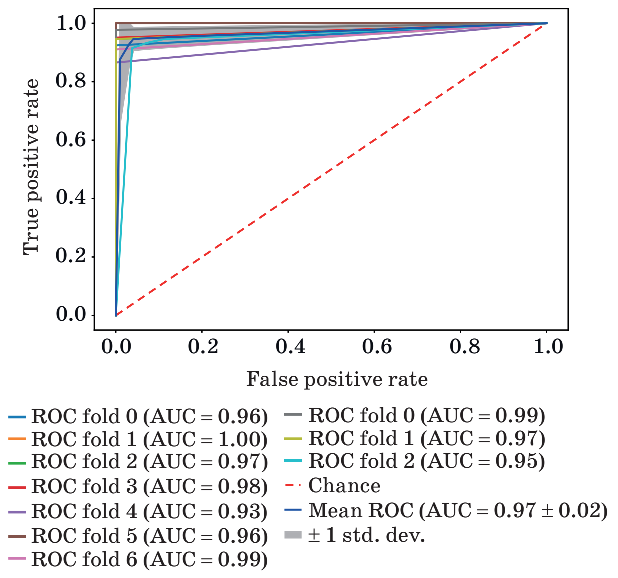
b)





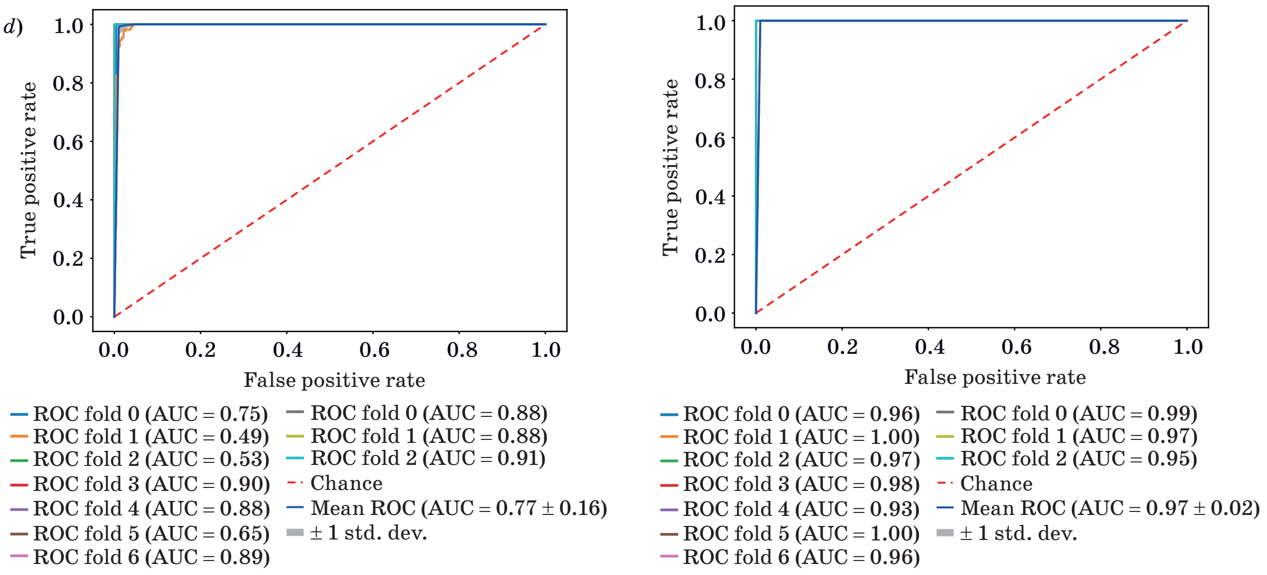
c)



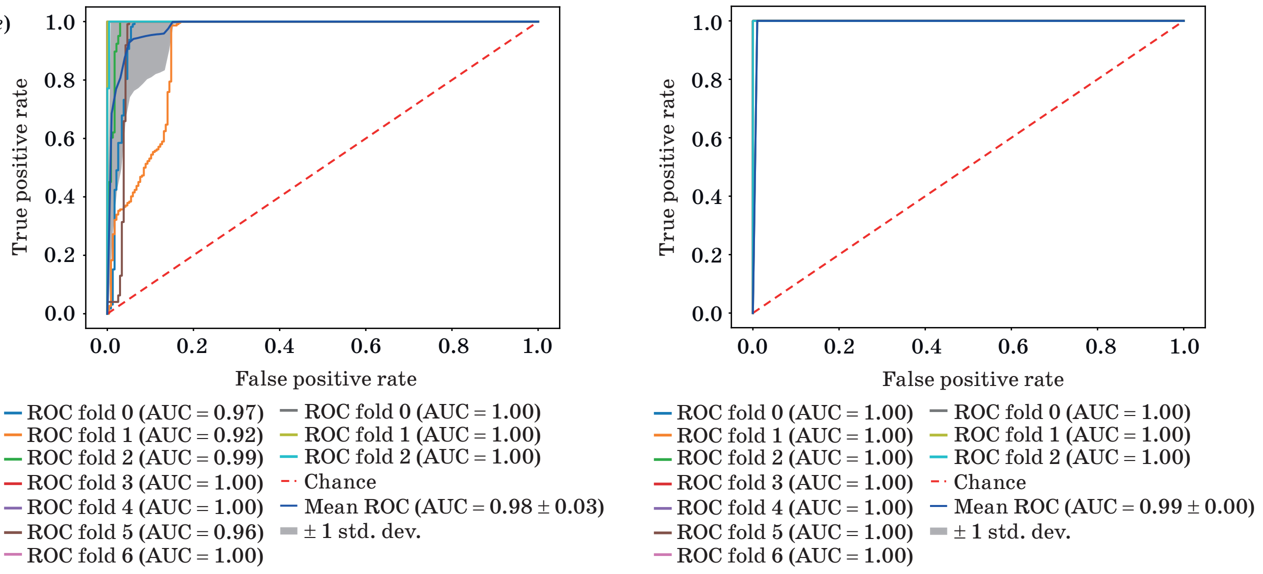


■ ***Fig. 6.*** ROC curve for DT-algorithm (*a*); RF-algorithm (*b*); KNN-algorithm (*c*); MLP-algorithm (*d*); SVM-algorithm (*e*);

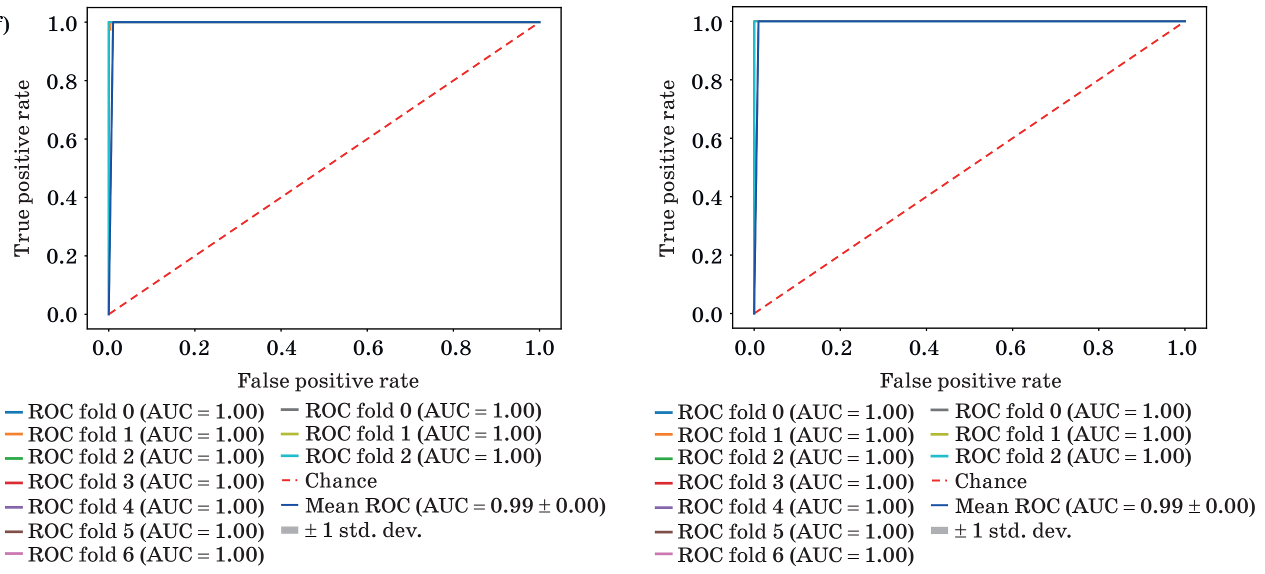
d)



e)



f)

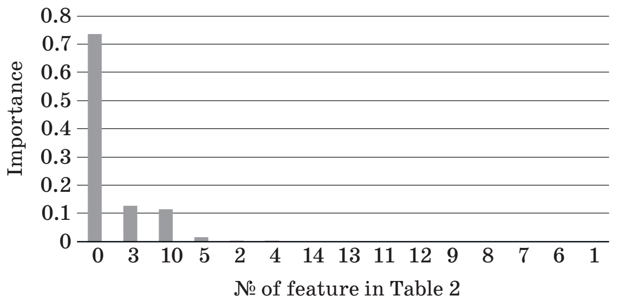


NB-algorithm (*f*) under bandwidth attack (left) and flow-table attack (on right)

In terms of attack type, the data shows that flow-table attack is a more recognizable type of attack than a bandwidth attack using the above algorithms.

We also try to check the features, which play a supporting role (important features) to the process of correctly classifying data for the DT-algorithm. Features that play the essential role are *byte_count*, *duration_sec*, *packet_count,* and some other features. It is also to be expected because, with DDoS attacks, the large amount of incoming packets makes *packet_count* change a lot compared to normal traffic's packet_count. Same with byte_count. However, most of the features that we collect from OpenFlow Switch are almost useless. The result is shown in Fig. 7.

Table 4 presents the features according to its importance for the detection of the anomalies.



■ ***Fig. 7.*** Importance of each feature

■ ***Table 4.*** Important features for classification

Position	Feature	Description
1st	Byte_count	Number of bytes in a flow
2nd	Duration_sec	Time flow has been alive in seconds
3rd	Packet_count	Number of packets in the flows
4th	Eth_src	Ethernet source address
5th	Duration_nsec	Time flow has been alive in nanoseconds
6th	Eth_dst	Ethernet destination address

Conclusion

The benefits of SDN network to overcome the drawbacks of a traditional network model are undisputed, but there are also certain limitations. For example, the entire system will collapse if the controller receives a DDoS attack and cannot respond to other valid requests. But based on what we have tested above and the results we have obtained, the application of ML-algorithms to detect these DDoS attacks is entirely possible, and gives very good results.

We tested six different ML-algorithms including RF, NB, KNN, SVM, MLP, and DT, to classify different types of traffic, including normal traffic, bandwidth attack traffic, and flow-table attack traffic. We have proved that DT, as well as NB, are suitable algorithms for DDoS attack detection (high accuracy and fast processing time, consume less resource compared to other algorithms).

Besides, we also pointed out that the main features that identify malicious traffic compared to normal traffic. It will make it easier to build a DDoS protection system with a more compact dataset, focusing only on the data needed.

Furthermore, we realized that flow-table attack is a more easily discovered attack than Bandwidth attack, as all six algorithms can predict this type with high accuracy. The efficiency of bandwidth attack detection is lower, so we need to focus more on this type to improve the predictive results.

In future work, we will focus on data quality criteria by comparing the results of detection between simulation dataset and real dataset.

References

1. Kreutz D., Ramos F. M. V., Verissimo P. E., Rothenberg C. E., Azodolmolky S., & Uhlig S. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 2015, no. 103(1), pp. 14–76. doi:10.1109/jproc.2014.2371999
2. Farhady H., Lee H. & Nakao A. Software-defined networking. *Computer Networks*, 2015, no. 81(81), pp. 79–95. doi:10.1016/j.comnet.2015.02.014
3. McKeown N., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S. & Turner J. OpenFlow: Enabling innovation in campus networks. *Acm Special Interest Group on Data Communication*, 2008, no. 38(2), pp. 69–74. doi:10.1145/1355734.1355746
4. Ahmad Ijaz, Namal S., Ylianttila M. & Gurtov A. Security in software defined networks: A Survey. *IEEE Communications Surveys and Tutorials*, 2015, no. 17(4), pp. 2317–2346. doi:10.1109/COMST.2015.2474118
5. Bawany N. Z., Shamsi J. A. & Salah K. DDoS attack detection and mitigation using SDN: Methods, practices, and solutions. *Arabian Journal for Science and Engineering*, 2017, no. 42(2), pp. 425–441. doi:10.1007/s13369-017-2414-5
6. Santos R., Souza D., Santo W., Ribeiro A. & Moreno E. Machine learning algorithms to detect DDoS at-

tacks in SDN. *Concurrency and Computation: Practice and Experience*, 2019, e5402. doi:10.1002/cpe.5402
7. Swami R., Dave M. & Ranga V. Defending DDoS against software defined networks using entropy. *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, 2019, pp. 1–5. doi:10.1109/IoT-SIU.2019.8777688
8. Yen T. F. & Reiter M. K. Traffic aggregation for malware detection. *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 2008, pp. 207–227. doi:10.1007/978-3-540-70542-0_11
9. Mehdi S. A., Khalid J. & Khayam S. A. Revisiting traffic anomaly detection using software defined networking. *International Workshop on Recent Advances in Intrusion Detection*, 2011, pp. 161–180. doi:10.1007/978-3-642-23644-0_9
10. Elsayed M. S., Le-Khac N., Dev S., & Jurcut A. D. Machine-learning techniques for detecting attacks in SDN. *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, 2019, pp. 277–281. doi:10.1109/ICCSNT47585.2019.8962519
11. Xiaoqiong X., Hongfang Y. & Kun Y. DDoS attack in software defined networks: a survey. *ZTE Communications*, 2017, no. 15(3), pp. 13–19.
12. Nam T. M., Phong P. H., Khoa T. D., Huong T. T., Nam P. N., Thanh N. H., Thang L. X., Tuan P. A. & Loi V. D. Self-organizing map-based approaches in DDoS flooding detection using SDN. *2018 International Conference on Information Networking (ICOIN)*, 2018, pp. 249–254. doi:10.1109/icoin.2018.8343119
13. Kokila R. T., Selvi S. T. & Govindarajan K. DDoS detection and analysis in SDN-based environment using support vector machine classifier. *2014 Sixth International Conference on Advanced Computing (ICoAC)*, 2014, pp. 205–210. doi:10.1109/icoac.2014.7229711
14. Yang L. & Zhao H. DDoS attack identification and defense using SDN based on machine learning method. *2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN)*, 2018, pp. 174–178. doi:10.1109/i-span.2018.00036
15. Deepa V., Sudar K.M. & Deepalakshmi P. Detection of DDoS attack on SDN control plane using hybrid machine learning techniques. *2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2018, pp. 299–303. doi:10.1109/icssit.2018.8748836
16. Braga R., Mota E. & Passito A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. *IEEE Local Computer Network Conference*, 2010, pp. 408–415. doi:10.1109/LCN.2010.5735752
17. Niyaz Q., Sun W., & Javaid A. Y. A deep learning based DDoS detection system in Software-Defined Networking (SDN). *ICST Transactions on Security and Safety*, 2017, no. 4(12), p. 153515. doi:10.4108/eai.28-12-2017.153515
18. Giotis K., Argyropoulos C., Androulidakis G., Kalogeras D. & Maglaris V. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. *Computer Networks*, 2014, vol. 62, pp. 122–136. doi:10.1016/j.bjp.2013.10.014
19. Javed A. & Latif S. Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques. *2014 National Software Engineering Conference*, 2014, pp. 55–60. doi:10.1109/nsec.2014.6998241
20. Dao N. N., Park J., Park M. & Cho S. A feasible method to combat against DDoS attack in SDN network. *2015 International Conference on Information Networking (ICOIN)*, 2015, pp. 309–311. doi:10.1109/icoin.2015.7057902
21. Nanda S., Zafari F., DeCusatis C., Wedaa E. & Yang B. Predicting network attack patterns in SDN using machine learning approach. *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2016, pp. 167–172. doi:10.1109/NFV-SDN.2016.7919493
22. Xie J., Yu F. R., Huang T., Xie R., Liu J., Wang C. & Liu Y. A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, no. 21(1), 2018, pp. 393–430. doi:10.1109/comst.2018.2866942
23. Bhuyan M. H., Bhattacharyya D. K. & Kalita J. K. *Network Traffic Anomaly Detection and Prevention: Concepts, Techniques, and Tools*. Springer, 2017. 263 p. doi:10.1007/978-3-319-65188-0
24. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V. & Vanderplas J. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011, no. 12, pp. 2825–2830.

УДК 004.056.5
doi:10.31799/1684-8853-2020-3-59-70

Сравнение алгоритмов машинного обучения при обнаружении DDoS-атак в программно-определяемых сетях

Д. Ч. Ле[a], канд. техн. наук, лектор, orcid.org/0000-0003-3735-0314, letranduc@dut.udn.vn
М. Х. Дао[a], магистрант, orcid.org/0000-0002-0998-6078
К. Л. Т. Нгуэн[a], магистр, лектор, orcid.org/0000-0003-4578-9925
[a]Университет Дананга — Университет науки и техники, факультет информационных технологий, 54, Нгуэн Лунг Банг, Дананг, 550000, Вьетнам

Введение: распределенная атака типа «отказ в обслуживании» (DDoS) стала популярным типом атак в кибербезопасности. Помимо обычных DDoS-атак, программно-определяемые сети сталкиваются с некоторыми другими типичными DDoS-атаками, такими как атака с использованием потоковой таблицы и атака контроллера. Одним из самых последних решений для обнаружения DDoS-атак является использование алгоритмов машинного обучения для классификации трафика. **Цель:** анализ применения алгоритмов машинного обучения для предотвращения DDoS-атак программно-определяемых сетей. **Результаты:** сравнение шести алгоритмов (случайный лес, дерево решений, наивный байесовский метод, машина опорных векторов, многослойный персептрон, k-ближайшие соседи) по критериям точность и время обработки показало, что дерево решений, как и наивный байесовский, являются лучшими алгоритмами для обнаружения DDoS-атак (высокая точность и быстрое время обработки, меньшее потребление ресурсов по сравнению с другими алгоритмами). Указаны и проанализированы основные функции, которые идентифицируют вредоносный трафик по сравнению с обычным трафиком: количество байтов в потоке, поток времени, Ethernet-адрес источника, Ethernet-адрес назначения. По результатам исследований сделан вывод, что атака с использованием таблицы потоков является более легкой для обнаружения, чем атака по пропускной способности. **Практическая значимость:** основные функции, которые играют вспомогательную роль в процессе правильной классификации данных, облегчают создание системы защиты от DDoS-атак с более компактным набором данных, включающим только необходимые данные. Алгоритмы, которые более подходят для машинного обучения, помогут точнее обнаруживать DDoS-атаки в программно-определяемых сетях.

Ключевые слова — DDoS, алгоритмы машинного обучения, атака по таблице потоков, атака по пропускной способности.

Для цитирования: Le D. T., Dao M. H., Nguyen Q. L. T. Comparison of machine learning algorithms for DDoS attack detection in SDN. *Информационно-управляющие системы*, 2020, № 3, с. 59–70. doi:10.31799/1684-8853-2020-3-59-70
For citation: Le D. T., Dao M. H., Nguyen Q. L. T. Comparison of machine learning algorithms for DDoS attack detection in SDN. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2020, no. 3, pp. 59–70. doi:10.31799/1684-8853-2020-3-59-70

ПАМЯТКА ДЛЯ АВТОРОВ

Поступающие в редакцию статьи проходят обязательное рецензирование.

При наличии положительной рецензии статья рассматривается редакционной коллегией. Принятая в печать статья направляется автору для согласования редакторских правок. После согласования автор представляет в редакцию окончательный вариант текста статьи.

Процедуры согласования текста статьи могут осуществляться как непосредственно в редакции, так и по e-mail (ius.spb@gmail.com).

При отклонении статьи редакция представляет автору мотивированное заключение и рецензию, при необходимости доработать статью — рецензию.

Редакция журнала напоминает, что ответственность за достоверность и точность рекламных материалов несут рекламодатели.