electronics

MDPI

Article

Machine Learning Approach Equipped with Neighbourhood Component Analysis for DDoS Attack Detection in Software-Defined Networking

Özgür Tonkal [1,*], Hüseyin Polat [2], Erdal Başaran [3], Zafer Cömert [1] and Ramazan Kocaoğlu [4]

[1] Department on Information Technology, Samsun University, 55080 Samsun, Turkey; zcomert@samsun.edu.tr
[2] Faculty of Technology, Gazi University, 06500 Ankara, Turkey; polath@gazi.edu.tr
[3] Department of Computer Technologies, Ağrı İbrahim Çeçen University, 04000 Ağrı, Turkey; ebasaran@agri.edu.tr
[4] Department on Computer Engineering, Ostim Technical University, 06500 Ankara, Turkey; ramazan.kocaoglu@ostimteknik.edu.tr
* Correspondence: ozgur.tonkal@samsun.edu.tr

Abstract: The Software-Defined Network (SDN) is a new network paradigm that promises more dynamic and efficiently manageable network architecture for new-generation networks. With its programmable central controller approach, network operators can easily manage and control the whole network. However, at the same time, due to its centralized structure, it is the target of many attack vectors. Distributed Denial of Service (DDoS) attacks are the most effective attack vector to the SDN. The purpose of this study is to classify the SDN traffic as normal or attack traffic using machine learning algorithms equipped with Neighbourhood Component Analysis (NCA). We handle a public "DDoS attack SDN Dataset" including a total of 23 features. The dataset consists of Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Control Message Protocol (ICMP) normal and attack traffics. The dataset, including more than 100 thousand recordings, has statistical features such as byte_count, duration_sec, packet rate, and packet per flow, except for features that define source and target machines. We use the NCA algorithm to reveal the most relevant features by feature selection and perform an effective classification. After preprocessing and feature selection stages, the obtained dataset was classified by k-Nearest Neighbor (kNN), Decision Tree (DT), Artificial Neural Network (ANN), and Support Vector Machine (SVM) algorithms. The experimental results show that DT has a better accuracy rate than the other algorithms with 100% classification achievement.

Keywords: SDN; Distributed Denial of Service attacks; Neighbourhood Component Analysis; machine learning

Citation: Tonkal, Ö.; Polat, H.; Başaran, E.; Cömert, Z.; Kocaoğlu, R. Machine Learning Approach Equipped with Neighbourhood Component Analysis for DDoS Attack Detection in Software-Defined Networking. *Electronics* **2021**, *10*, 1227. https://doi.org/10.3390/electronics10111227

Academic Editors: Houbing Song and Jehad Ali

Received: 24 April 2021
Accepted: 19 May 2021
Published: 21 May 2021

1. Introduction

SDN is a new paradigm that facilitates network management with its dynamic and programmable structure. In SDN, control and data planes are divided from each other, and network management is carried out by a central controller [1]. Thus, the controller, which can manage the whole network from a single point, can quickly apply different network policies to the whole network. Figure 1 shows the layered structure of the SDN environment. However, this emerging new approach brings along security problems in addition to the advantages it provides. In addition to attacks encountered in traditional network structures, SDN is also exposed to attacks specific to itself [2]. Perhaps the most dangerous of these attacks are attacks on the controller, because the attacker who seizes the controller can have the ability to manage or disrupt all network traffic. DDoS attacks in which users are denied access to network services are at the top of the attacks on the controller.

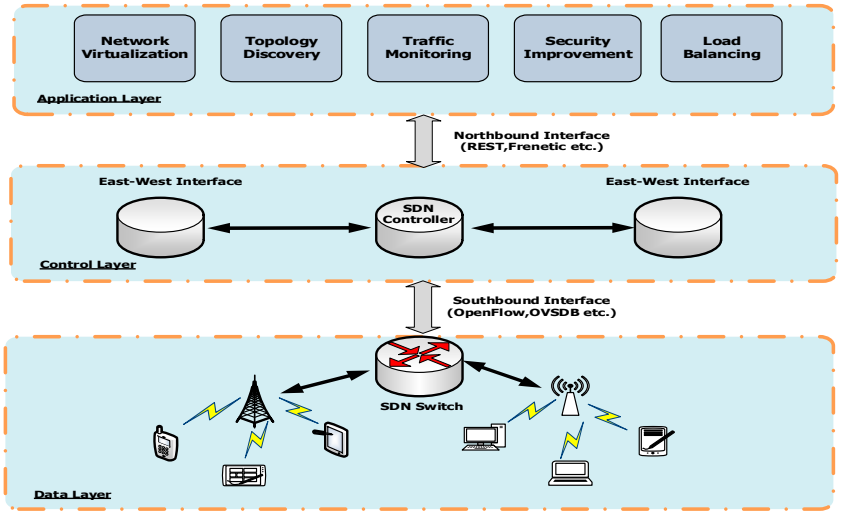

Figure 1. SDN architecture.

The attackers aim to create heavy traffic with more than one machine, to consume the resources on the target machine, and to prevent it from serving after a while by DDoS attacks. Attackers use "botnets" created from devices called zombies hijacked by internet hackers. DDoS attacks are carried out with a large number of machines, so it is very difficult to detect and block. The frequency and severity of DDoS attacks are constantly increasing and can have fatal effects on many network services [3,4]. For this reason, quick detection and prevention of DDoS attacks are some of the most important problems for network service providers and administrators. Different SDN layers can be disabled by filling communication channels between the controller and the switch or between the controller and the application layer with unnecessary flow information by DDoS attacks. There is no built-in security mechanism on the controller that can distinguish between attack traffic and normal traffic. Therefore, it is very difficult to detect an attack.

DDoS attacks are grouped into three categories; application-layer attacks, resource-consuming attacks, and volumetric attacks [5]. Application-layer attacks consist of complex attacks. They target specific services using less bandwidth and slowly consume network resources. Therefore, it is difficult to detect. Hypertext Transfer Protocol (HTTP) and Domain Name System (DNS) attacks can be evaluated in this category [6]. In resource-consuming attacks, servers are rendered unavailable by taking advantage of vulnerabilities in protocols implemented on the network layer. TCP-SYN Flood consumes the resources of the target machine (memory, CPU, and storage) [7]. It aims to consume the bandwidth of the network with volumetric attacks. Common attacks such as ICMP, UDP, and TCP-SYN flood are performed by using vulnerabilities in Layer 3 and Layer 4 protocols [8].

In this study, we focus on the SDN to ensure a lightweight hybrid model equipped with NCA and machine learning approaches to contribute to ensuring a new-generation manageable network architecture. In detecting DDoS attacks with machine learning, some flow characteristics (packet size, arrival time, response time, packet rate, packet per flow, etc.) are used to identify whether the network traffic is normal. DDoS attacks often use the same average packet size. Since the attack traffic has a high bitrate, the time to arrive at the target machine is very short. Attackers focus on any of these features to consume the target machine's resources and prevent it from serving. For this purpose, we handle a public dataset including a total of 23 features for detecting DDoS attacks with machine learning. Instead of considering all the features in the dataset, we reveal the most efficient features with the NCA approach with the help of the newly proposed model. To ensure more generalized results, the proposed approach is tried and tested in four different machine learning algorithms. As a result, the obtained promising results point out that the proposed approach can achieve more efficient results compared to traditional machine learning

algorithms, even while using fewer features. The proposed model has great potential in contributing to the management of new-generation SDN architecture.

The rest of the paper is organized as follows: The next section elaborates on some previous related works. In Section 3, information about the used publicly available dataset is briefly given. in addition, the existing models, feature selection method, data augmentation method, machine learning method, optimization method, and the proposed method are presented briefly in this section. The results and analysis are given in Section 4. The discussion is presented in Section 5. Finally, Section 6 includes the concluding remarks and future work.

2. Related Works

In recent years, many studies have been done to secure SDN using machine learning techniques. In this section, we discuss several studies of DDoS security mechanisms based on machine learning and deep learning techniques.

Security solutions such as the Intrusion Prevention System (IPS) and the Intrusion Detection System (IDS) are used to ensure network security. The increasing variety of attacks has made it necessary to make statistical calculations on these systems. With machine learning algorithms, IDS systems have gained the ability to make meaningful comments and predictions. Pérez-Díaz et al. [9] proposed a new architectural solution to detect Low-Rate DDoS (LR-DDoS) attacks and mitigate their effectiveness in SDN. The architectural solution consists of IPS and IDS modules placed on the controller. Attack detection is made using different trained machine learning and deep learning methods through the Identification Application Programming Interface (API) positioned in the IDS module. They used the Canadian Institute of Cybersecurity (CIC) DoS dataset in their studies. The experimental results showed that the algorithm that gives the best result with 95% accuracy among six different machine learning algorithms is Multi-Layer Perceptron (MLP). Shoo et al. [10] introduced a new evolutionary model to classify DDoS attack traffic in an SDN environment. The model uses a combined SVM algorithm for malicious traffic classification. Genetic algorithms (GA) were used for SVM optimization when determining Kernel Principal Component Analysis (KPCA) as a property-selection method to improve the model's classification performance. Two different datasets which consist of UDP flood, HTTP flood, Smurf, SiDDoS and normal traffics were used to test and compare model accuracy. The experimental results show that the proposed combined method accuracy is 98.9%.

Kyaw, Aye Thandar, May Zin Oo, and Chit Su Khin [11] used two machine learning algorithms to detect UDP flooding attacks in the SDN environment. They used the Scapy tool for traffic packet generation. Their system collects the flow statics via the OpenFlow switch. After the feature extraction phase, they compared the classification performance of Linear and Polynomial SVM models. Experimental results show that the Polynomial SVM algorithm has a 34% lower false alarm rate with 3% better accuracy.

Janarthanam, S., N. Prakash, and M. Shanthakumar [12] proposed the security framework that detects DDoS attacks on the SDN environment. The framework is based on an adaptive learning model that uses the historical dataset for traffic classification. They used a cross-validation approach for efficient classification results. Although the results obtained are promising, the adaptive security model should be tested on different datasets obtained from the real environment to be more realistic. Tan, Liang et al. [13] proposed a novel security model for DDoS attacks in the SDN environment. The model involves two modules based on ML algorithms. The data-processing module uses the K-Means algorithm for best feature selection and the detection module uses the *k*-nearest neighbour (*k*NN) algorithm to detect attack flows. Compared to the distributed-Self-Organizing Map (SOM) and entropy-based method, their method has a 98.85% accuracy with a 98.47% recall rate.

Wang, Lu, and Ying Liu [14] proposed a DDoS attack detection method that used a two-level detection system to identify the attack based on information entropy and deep

learning. They used entropy detection to detect suspicious traffic at the first level, and at the second level, they used the convolutional neural network (CNN) model to detect attack traffic. Finally, they tested the method using deep neural networks, decision trees (DT), and SVM models. The CNN model's accuracy was 4.25–8.20% higher than the other algorithms.

Deepa, V., K. Muthamil Sudar, and P. Deepalakshmi [15] proposed an ensemble technique to detect denial of service (DDoS) attacks. They used four different machine learning models to detect suspicious traffic in the SDN environment. SVM-SOM algorithm showed better results compared to the other ML algorithms with 98.12% accuracy. The authors in [16] introduced a DDoS attack-detection system for SDN. The system used two security stages. Firstly, they used Snort to detect signature-based attacks. After that, they used the SVM classifier and the deep neural network (DNN) machine learning model for attack classification. The experimental results proved that DNN has a better classification accuracy rate than SVM at 92.30%.

The authors of [17] demonstrated the success of the deep learning model in detecting and classifying DDoS attacks in their studies. They applied the DNN model on two different samples taken over the CICDDoS2019 dataset. The attack detection scenario was applied on the first dataset, while the attack traffic classification scenario was applied on the second dataset. Their results showed that the DNN model is quite successful in both intrusion detection and classification. The authors generalize on the results they obtained on the CICDDoS2019 dataset in their studies. However, different datasets can give different results. Therefore, they could support their work by working on different datasets such as NSL-KDD, ISCX IDS 2012, UNSW-NB15, and CICIDS 2017.

Some of the researchers have made intrusion detection using hybrid machine learning models in their studies. Nam, Tran Manh et al. [18] proposed a DDoS security system using the SDN architecture to detect attack flows. Their hybrid solution uses combined kNN and SOM algorithms. They classified the traffic into normal and malicious using flow statistics collected from SDN switches and vehicle sensors. Adhikary et al. [19] focused on a hybrid technique which was combined the technique of Neural Network and DT for different types of DDoS attacks in Vehicular Ad hoc Network (VANET). The proposed hybrid algorithm has better results than the single models of Neural Networks and DT. Hosseini and Azizi [20] proposed a hybrid model to detect and mitigate the DDoS attack. Their framework separated the sides as proxy and client. This way, the limited resources on both sides can be used effectively. They combined six different ML techniques to identify the attack flows. Random Forest classifier provides better results than the compared ML techniques.

Several machine learning-based solutions to detect DDoS attacks in cloud computing and IoT networks have been proposed. The big challenge in machine learning-based solutions is the detection of these attacks with high accuracy. Ujjan, Raja Majid Ali et al. [21] focused on Internet of Things (IoT) DDoS attack detection. Their proposed methods used time-based and packet-based sampling approaches to collect network traffic coming to the SDN data plane. With these sampling approaches, they aim to reduce the IDS and Deep Neural Network (DNN) model's processing load and increase the classification performances. The results show that their proposed model has higher detection rates. Ravi, Nagarathna, and S. Mercy Shalinie [22] proposed a security mechanism to detect DDoS attacks mitigation in the IoT networks. Their mechanism, which is named Learning-driven Detection Mitigation (LEDEM), used a semi-supervised ML model for malicious traffic detection. LEDEM has multiple customized controllers connected to a central controller. They have implemented different security approaches for IoT environments that they separate as mobile IoT and Fixed IoT. They used their dataset for testing their security mechanism. Yong et al. [23] focused on the web-shell intrusion in IoT environment using the ensemble methods. The authors used the principal component analysis to select the best features with three types of ensemble techniques: random forest (RF), voting, and

extremely randomized trees (ET). While RF and ET work well for the light IoT environment, the voting method gives better results for heavy IoT scenarios.

The authors of [24] proposed a machine learning-based DDoS intrusion detection system to ensure the security of cloud services. They developed the Self-Adaptive Evolutionary Extreme Learning Machine (SaE-ELM) model as an automatic adaptive system and applied it to intrusion detection systems. They tested their method on four different datasets and compared the classification accuracy with commonly used machine learning models such as ANN, DT, and SVM. Although the test and training time of the model they developed is slightly higher compared to the SaE-ELM model, the results obtained are quite good.

Although the central management and programmable structure provided by SDN brings new capabilities to IDSs, the performance of these detection systems depends on the quality of training datasets.

In the recent studies we have summarized above, different datasets such as KDD Cup'99, NSL-KDD, CICIDS2017, CAIDA 2016, UNB-ISCX, and CIC DoS were used. The biggest problem with these datasets is that they are out of date. Attack characteristics are changing, so the need for up-to-date datasets is increasing. LITNET-2020 dataset [25] and Boğaziçi University datasets [26] are the current datasets used to detect DDoS attacks. However, these datasets are also created using traditional network platforms like the other datasets.

Our motivation in this study has been to work on up-to-date datasets obtained from SDN network platforms. There are a few publicly available datasets that can be used directly for anomaly detection systems applied in SDN networks [27,28]. We used the "DDOS attack SDN Dataset" in our study, which is also a new dataset and accessible to researchers for use in machine learning and deep learning research.

3. Materials and Methods

The steps of the method followed to achieve the results are given in Figure 2. Furthermore, the features and classes of the public dataset used in this section are explained. The feature selection algorithm used to determine the features that will increase the classification accuracy in the dataset is given, and the features of the classifiers used after the feature selection phase are also explained in detail in this section.

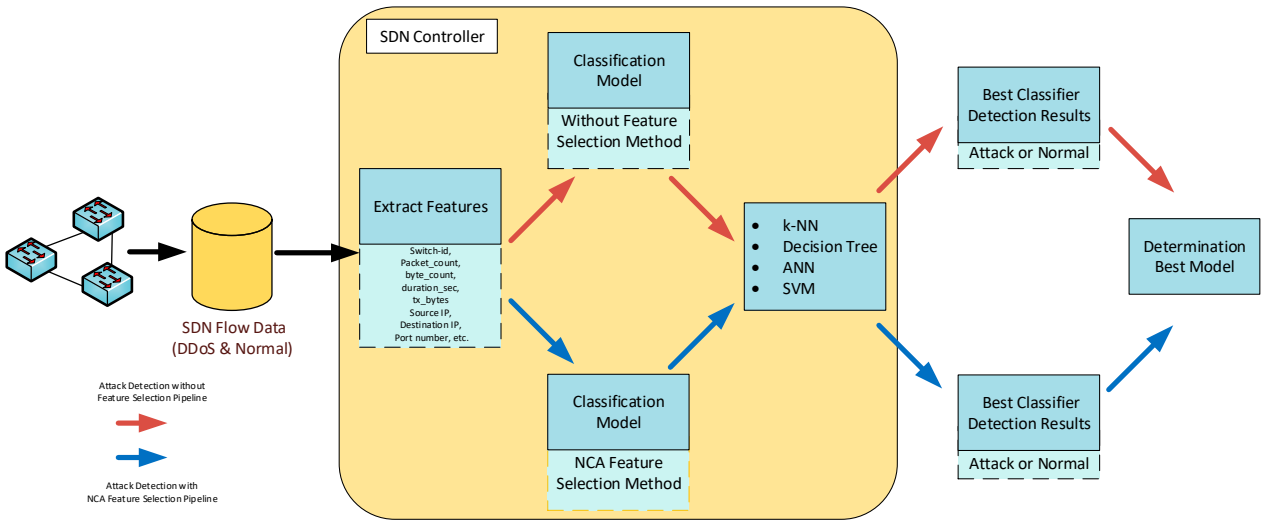

Figure 2. The steps of the proposed model.

3.1. Dataset

In this study; the public "DDOS attack SDN Dataset", which was created in the SDN environment and made publicly accessible to researchers for use in machine learning and deep learning research, was used [29]. There are 23 features in the dataset, which contains 104345 traffic flows. The dataset consisting of TCP, UDP, and ICMP traffic is shown using the normal and attack traffic class label. The dataset has statistical features such as byte_count, duration_sec, packet rate, and packet per flow, except for features that define source and target machines. Before starting machine learning model training, the data must be preprocessed. At this stage, packet rate, byte per-flow, and packet per flow properties are excluded from the dataset because they contain duplicate values. Categorical variables such as source-destination IP and protocol that do not have numeric values were encoded by using one-hot encoding [30]. In the next step, we tried to find the correlation of input features with output features by using various machine learning methods, heatmap graph, and correlation techniques. As a result of this process, the column shown with the "dt" feature and containing the time information was found to be unnecessary and removed from the dataset. The data preprocessing phase was terminated by applying normalization to numeric data.

3.2. Machine Learning Approach

Machine learning methods are used to analyze system performance and detect unusual events that are not consistent with normal network behaviour. Especially in network systems where high-density data is circulating, abnormal movements are detected by mathematical models created using machine learning algorithms, and preventive policies are quickly applied to network systems. In this section, the features of the machine learning approaches used in this study are briefly explained.

3.2.1. *k*-Nearest Neighbor (k-NN) Classifier

k-NN is one of the popular machine learning algorithms. It is a non-parametric, distance-based, and supervised approach that was introduced in 1951 [31]. This algorithm measures the similarities in the dataset considering a distance function. The test data are classified based on the majority votes of its *k*-nearest neighbours.

A training set is defined as X and Y pairs. Let $X = \{x_1, x_2, \ldots\ x_n\}$ where $x_i \in R^n$ corresponds to the training data in the n-dimension feature set, and let $Y = \{y_1, y_2, \ldots\ y_n\}$ match the target labels. A prediction for a test data $\hat{x}$ applied as input to the *k*NN model is realized as follows [32]:

- A distance function such as a Euclidean one is used to measure similarity in the training data. For two points named a and b have Cartesian coordinates $(a_1,\ a_2)$ and $(b_1,\ b_2)$, the distance between a and b is calculated as given in Equation (1):

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \quad (1)$$

- The label of test data $\hat{x}$ is determined considering the majority votes of its k-nearest neighbours.

3.2.2. Decision Tree

The decision tree machine learning algorithm is used for regression as well as the classification of real-world problems. This model is inspired by a tree structure. However, the root of the tree is located at the top. The branches are created considering objective rules relied on the features of the dataset, and the decision tree is also progressively developed [33]. To create a decision tree, the procedures described below can be followed [34]:

1. The whole dataset is divided into two parts as training and test sets.
2. The training set is applied as the input to the root of the tree.
3. The root is determined using information theory as described in Equation (2).

4. The prone procedure is carried out.
5. The procedures between 1 and 4 are followed again and again until all nodes become leaf nodes.

$$Entropy(P) = -\sum_{i=1}^{N} p_i \log(p_i) \quad (2)$$

where the probability distribution of the dataset is denoted with p. To achieve an efficient decision tree, there are also other hyper-parameters, such as the minimum leaf size, minimum parent size, and the maximum number of splits to be set.

3.2.3. Artificial Neural Network

ANN is a useful computational model for making predictions on nonlinear and complicated systems. In the basic approach, an ANN model consists of an input layer, one or more hidden layer(s), and an output layer [35]. This computational model can be defined as in Equation (3)

$$o_i = \sigma\left(\sum_{j=1}^{N} \omega_i x_i + b^i\right) \quad (3)$$

Herein, the weights of the model are denoted with ω_i, and these weights are calculated using training algorithms. The data describing the problem with features is shown by x_i, and bias values are symbolized by b^i [36].

In the configuration stage of the ANN model, we used three hidden layers with 25, 15, 10 computational nodes, respectively. Levenberg–Marquardt training algorithm was preferred. Other hyper-parameters were used with their default values.

3.2.4. Support Vector Machine

The SVM algorithm is one of the most efficient machine learning algorithms for classification and regression problems [37]. SVM determines a hyperplane that can separate the space into two or more classes. The margin is kept large as possible, and the data points in this border are called support vectors [38]. The kernel is used to divide the data non-linearly in SVM. To this aim, the SVM searches support vectors, weights, and bias. For input data, $z \in R^n$, the SVM is determined as follows:

$$f(z) = sign\left(\sum_{i=1}^{N} v_i \Psi(z_i) + c\right) \quad (4)$$

Herein, $\Psi(.)$ corresponds to the mapping function, and v and c are weights and bias, respectively. The mapping function can be linear SVM, polynomial SVM, radial-basis function (RBF)-SVM. In this study, we preferred RBF-SVM as a mapping function for the classification task.

3.3. Neighbourhood Component Analysis

The main purpose of feature selection can be expressed as the selection of a subset feature by reducing the cost of computing and reducing unrelated features from the feature set that will affect model performance [39]. In this study, the NCA algorithm was used to select the most appropriate features to perform an effective classification of more than 100 thousand network records, which consists of 22 features of SDN technology. The advantage of the NCA model developed based on the kNN algorithm is that it lists the features in order of importance and also provides information about the weight value of the features [40,41].

There is a possibility that a feature given as x_i input in the NCA algorithm corresponds to y_i the class, corresponding to all classes. The distance between two observations is calculated according to Equation (5) [41].

$$d_w = \sum_{r}^{p} w_r^2 |x_{ir} - x_{jr}| \quad (5)$$

Here, w_r is the weight value of the feature. The reference points (P) in the feature set are calculated according to Equation (6).

$$P\left(Ref(x_i) = x_j | S\right) = \frac{k\left(d_w\left(x_i, x_j\right)\right)}{\sum_{j=1}^{N} k\left(d_w\left(x_i, x_j\right)\right)} \tag{6}$$

The probability of choosing x_i as the reference point for xj is calculated according to Equation (7).

$$p_{ij} P\left(Ref(x_i) = x_j | S\right) = \frac{k\left(d_w\left(x_i, x_j\right)\right)}{\sum_{j=1\ j \neq i}^{N} k\left(d_w\left(x_i, x_j\right)\right)} \tag{7}$$

Herein, k corresponds to the kernel function ($k(z) = (\exp -z/\sigma)$) and σ denotes the width of kernel function whereas the correct classification possibility of the real class is calculated as defined in Equation (8) [42].

$$p_i = \sum_j y_{ij} p_{ij} \tag{8}$$

where y_{ij}= 1 if and only if $y_j = y_j$ and y_{ij}= 0.

3.4. Performance Metrics and Model Evaluation

A confusion matrix was used to test the performance results of the experimental studies conducted to determine the normal and abnormal network records obtained with SDN. This matrix contains estimated values and real values. The confusion matrix is given in Table 1. Here, true positive (TP) and true negative (TN) represent the correctly predicted values of network movements, while false positive (FP) and false negative (FN) represent incorrect predicted values [43,44]. Moreover, the receiver operating curve (ROC) and the areas under the curves (AUC) were used to evaluate the model performance. ROC curve has a false positive rate on the horizontal axis and a true positive rate on the vertical axis.

Table 1. Confusion Matrix.

	True Class	
Predicted Class	TP	FP
	FN	TN

The proposed model was evaluated based on the accuracy (Acc), sensitivity (Se), specificity (Sp), Precision (Pr), and F-score performance metrics derived from confusion matrices. The formulations of these metrics are given in Table 2. In addition, the k-fold cross-validation method was used in this experimental study to determine the test error of the predictive model. In the cross-validation method, the dataset is divided into k groups and it is a less biased model [38]. The k value was determined as 10 in this experimental study.

To carry out experimental studies, more than 100 thousand network records consisting of 22 different features of network movements were kept. Before the feature selection and classification phase of the network records, the preprocessing phase was carried out. The standardization process, which is widely used in the field of machine learning, was applied as preprocessing. Standardization can be expressed as the recalculation of variance according to the defined value [45]. Mathematically, it is calculated as in Equation (9).

$$S_i = (S_i - \mu_i)/\sigma_{\mu_i} \tag{9}$$

In Equation (9), S_i refer to raw data while μ_i and σ_{μ_i} indicate mean and standard deviation, respectively.

Table 2. Performance metrics with definitions and formulas.

Metric	Formulation	Definition
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	The overall accuracy of the model.
Sensitivity	$\frac{TP}{TP+FN}$	The performance of the model on detecting abnormal network traffic.
Specificity	$\frac{TN}{TN+FP}$	The performance of the model on detecting normal network traffic.
Precision	$\frac{TP}{TP+FP}$	The ratio of correctly predicted abnormal network traffic to the total abnormal network traffic.
F-Score	$\frac{2*TP}{2*TP+FP+FN}$	The accuracy of the model on the whole dataset.

4. Experimental Results

In the first stage of the experimental study, SDN records were classified directly with machine learning methods after the preprocessing step without any feature selection. Hyper-parameters of machine learning algorithms were determined automatically using the method of optimization of hyper-parameters to perform an effective classification. While the dataset was divided as training at the rate of 0.7, it was separated as a test at the rate of 0.3. To perform the classification process with the *k*NN algorithm, the value of *k*, which is the number of neighbours to be looked at, was determined as 1, and Euclidean was chosen as the distance function. The Gini algorithm is determined as the division criterion in the DT method. The hidden neuron number was 10 for classification with the ANN method, and the Levenberg-Marquardt algorithm was used as the training algorithm. To classify the network records with the SVM method, the kernel Radial Basis Function was selected, the box constraint value was determined as 1, and the kernel scale value was determined as 0.9. When the classification results were examined after the SDN records were given to the input of the machine learning algorithms, the best accuracy rate was obtained with the ANN method at 97.35%, while the accuracy rates of 95.41%, 94.14%, and 80.56% were obtained with kNN, DT, and SVM methods, respectively. Performance results obtained as results of classification are given in Table 3.

Table 3. Classification Results of ML Model without Feature Selection Method.

ML	Acc (%)	Se (%)	Sp (%)	Pr (%)	Fsc (%)
*k*NN	95.41	96.95	93.02	95.58	96.26
DT	94.14	90.38	100.0	100.0	94.95
ANN	97.35	95.55	98.55	97.78	96.65
SVM	80.56	87.17	70.26	82.04	84.53

The raw network data in the dataset were carried out in the preprocessing step, and feature selection was applied with the NCA algorithm. To train the NCA algorithm, the regularization parameter value lambda (λ), which prevents overfitting, was automatically determined. The stochastic gradient descent (SGD) method was used to optimize feature weights. In SGD optimization, the mini-batch size value was determined as 10 and the epoch value as 5. While the weight values of the unrelated features in the NCA algorithm are close to zero, the weight values of the features with high discrimination features are higher. The index values of features and the corresponding weight values are given in Figure 3.

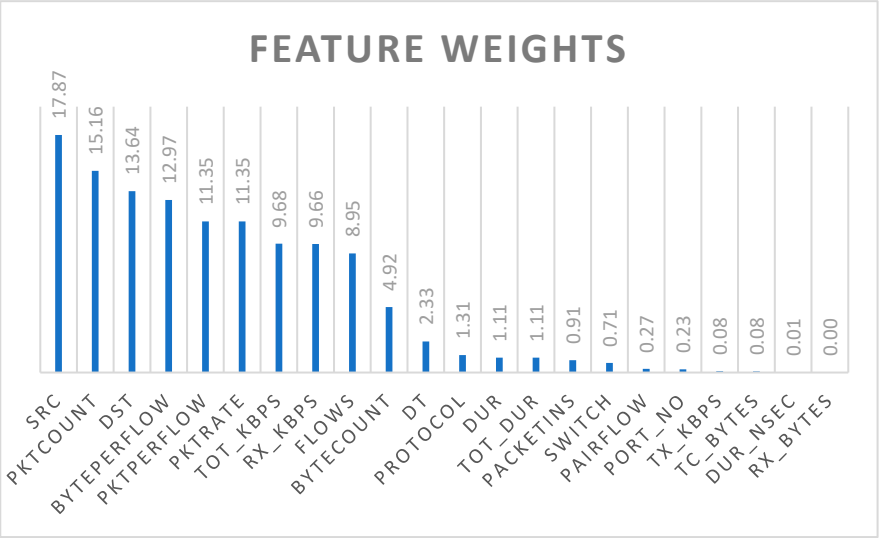

Figure 3. NCA filtering of network movement features.

When we look at the weight values of the features with the NCA algorithm, it is observed that the weight values of eight features are between 0 and 1, while the weight values of 14 features vary between 1.11 and 17.87. Machine learning algorithms are known to affect computational costs when classifying high-specification problems [46]. For this reason, after analyzing 22 network features NCA algorithms, the first classification process was made with eight features with an index value of more than 9. In the second experimental study, 14 effective features were selected and given as input data to machine learning algorithms. The 14 most effective properties and weight values selected by NCA are given in Table 4.

Table 4. The 14 most effective features and weights selected by NCA.

Features	NCA Feature Weight
src	17.87
pktcount	15.16
dst	13.64
byteperflow	12.97
pktperflow	11.35
pktrate	11.35
tot_kbps	9.68
rx_kbps	9.66
flows	8.95
bytecount	4.92
dt	2.33
protocol	1.31
dur	1.11
tot_dur	1.11

More than 100 thousand network records were classified by *k*NN, DT, ANN, and SVM algorithms after preprocessing and feature selection. In the first experimental study, the new dataset, created by selecting the features with an index value of more than 9 with the NCA algorithm, is given as an input to ML algorithms. As a result of experimental studies, promising results were obtained with all classification algorithms. While the best accuracy rate was obtained with the DT method as 99.1760%, it was determined as 97.7542%, 96.2015%, and 81.4810% with the *k*NN, YSA, and SVM methods, respectively. The performance results obtained by machine learning methods as a result of the experimental study with the most efficient eight features are given in Table 5. ROC curves of the whole machine learning method with eight features are given in Figure 4.

Table 5. Classification results of ML models with the most efficient eight features.

ML	Acc (%)	Se (%)	Sp (%)	Pr (%)	Fsc (%)
*k*NN	97.7542	97.4425	98.259	97.7617	98.1704
DT	99.1760	99.8780	98.1199	99.1802	99.3223
ANN	96.2015	94.6904	97.1788	96.2107	95.1412
SVM	81.4810	82.8702	78.9583	81.3412	85.2325

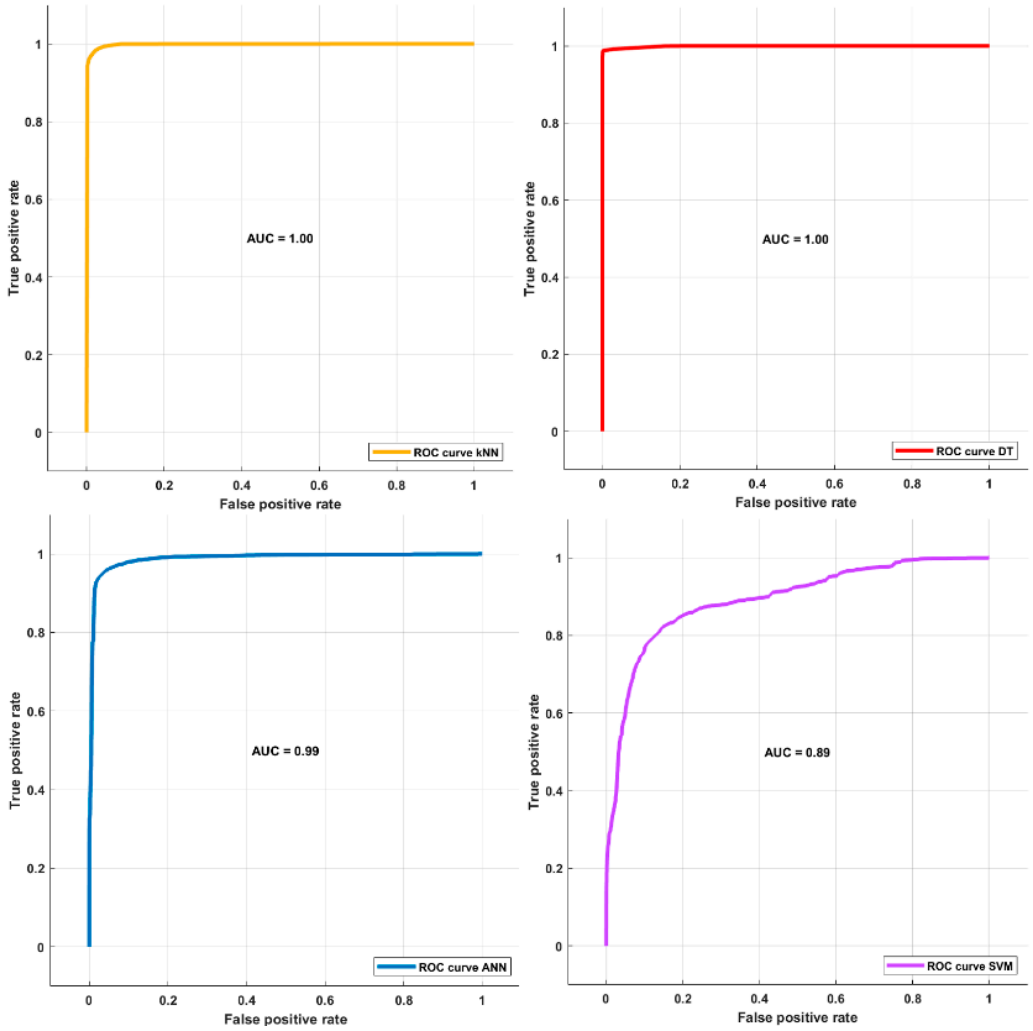

Figure 4. ROC curves of all ML models with 8 features.

In the second experimental study, the feature set consisted of 22 features. After training with the NCA algorithm, the features with an index value of more than 1.11 were selected. They were classified by ML methods using the same hyperparameters as in the first experimental study. As a result of experimental studies, very good results were obtained with all classification algorithms. While the best accuracy rate was obtained as 100% with the DT method, it was determined as 99.15%, 99.78%, and 98.59% with *k*NN, ANN, and SVM methods, respectively. The performance results obtained by machine learning methods as a result of the experimental study are given in Table 6. In Figure 5, the ROC curves of all machine learning methods are given.

Table 6. Classification results of ML models.

ML	Acc (%)	Se (%)	Sp (%)	Pr (%)	Fsc (%)
*k*NN	99.1502	99.1068	99.2187	99.1068	99.3039
DT	100.0	100.0	100.0	100.0	100.0
ANN	99.7834	100.0	99.6444	100.0	99.7237
SVM	98.5935	98.6509	98.5032	98.6509	98.8473

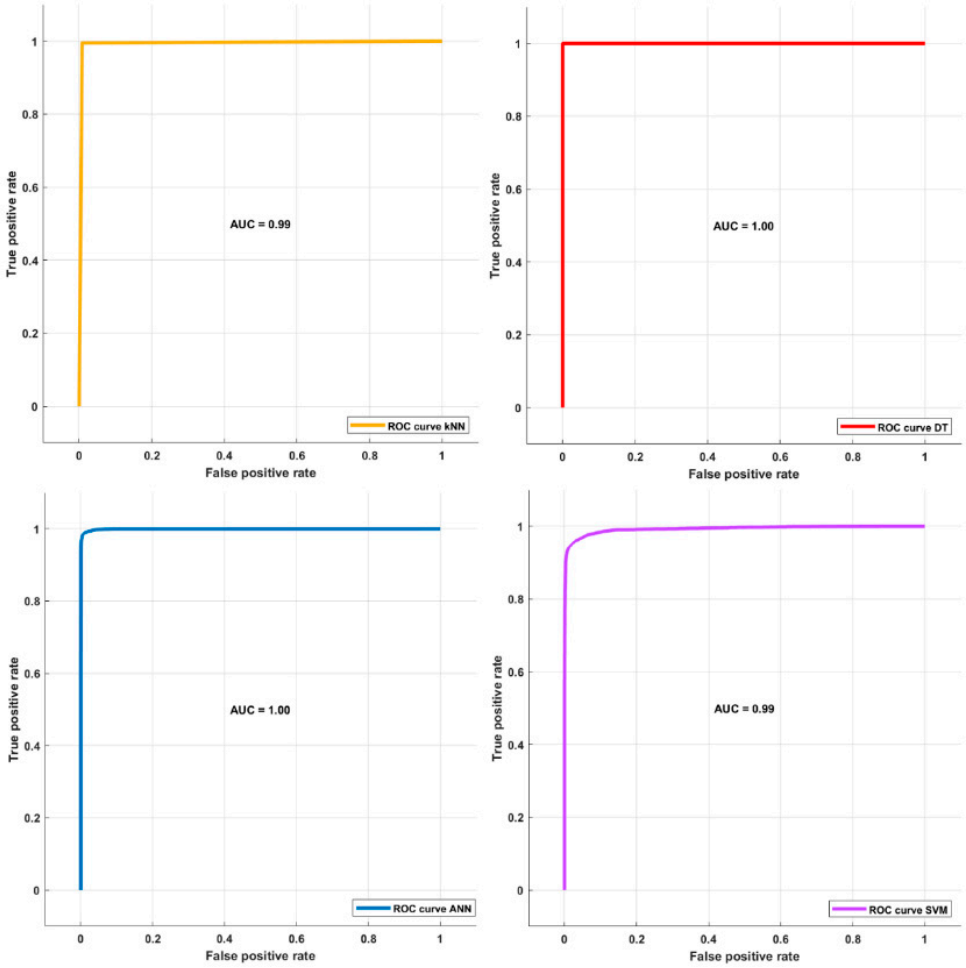

Figure 5. ROC curves of all ML models.

Finally, the feature set consisting of 22 features showed the best results and an index value above 1.11. It was also subjected to a cross-validation test. As a result of the experimental study, the highest accuracy rates were obtained by the DT method. While the accuracy rate was 99.82% with the DT method, it was determined as 99.23%, 97.63%, and 97.20% with the *k*NN, ANN, and SVM methods, respectively. Performance values obtained by cross-validation test are given in Table 7.

Table 7. 10 fold-cross validation results.

ML	Acc (%)	Se (%)	Sp (%)	Pr (%)	Fsc (%)
kNN	99.23	99.62	98.64	99.13	99.37
DT	99.82	99.77	99.91	99.94	99.85
ANN	97.69	97.86	97.43	98.34	98.10
SVM	97.20	97.52	96.70	97.87	97.70

5. Discussion

In Table 8, the studies on DDoS attack traffic detection using machine learning algorithms and the classification model we propose are shown comparatively. When Table 8 is examined, it is seen that different datasets were used to detect attack traffic. Some of the researchers used public datasets containing network traffic data from conventional network topologies such as KDD Cup'99, NSL-KDD, UNB-ISCX, CICIDS2017, and CAIDA 2016 [2–8]. The use of these datasets is positive for comparing the performance of machine learning algorithms used in the detection of attack traffic. However, the fact that the SDN architecture is different from the conventional network architecture causes SDN to have unique attack vectors other than its current attacks. Furthermore, the increasing number of attack traffic and variety requires the use of up-to-date datasets. For this reason, researchers use their datasets obtained by using the SDN architecture for their work [11,12,14,19–21,27,28]. The SDN-specific dataset used in this study was created by the Bennett University study group for machine learning and deep learning studies. The most important criterion for selecting the dataset is that it is created using the SDN architecture and includes up-to-date SDN DDoS traffic data.

Table 8. The comparison of the related studies.

Related Studies and Datasets	Feature Selection	ML Algorithms	Accuracy (%)
CIC DoS dataset [9].	Without feature selection	Random Tree, J48, REP Tree, SVM, Random Forest, MLP	95.00
NSL-KDD [10].	KPCA	SVM	98.91
NSL-KDD [13].	Without feature selection	K-Means and kNN	98.85
Their dataset [21].	Without feature selection	Stacked Autoencoders (SAE) deep learning model	95.00
UNB-ISCX [22].	Without feature selection	Semisupervised machine-learning algorithm	96.28
Their dataset [11].	Without feature selection	Polynomial SVM- Linear SVM	95.00
CICIDS2017 [14].	Without feature selection	CNN	98.98
Their dataset [12].	Without feature selection	ALM	97.00
CAIDA 2016 [15].	Without feature selection	kNN, Naive Bayes, SVM, and SOM	98.12
KDD Cup'99 [16].	Without feature selection	SVM classifier and DNN	92.30
CAIDA "DDoS Attack 2007" [18].	Entropy-based selection	SOM+kNN, SOM distributed-center	98.24
Their dataset [19].	Without feature selection	Hybrid algorithm of DT and Neural Network	96.40
NSL-KDD, the introduced dataset in [14,20].	KNIME forward feature selection	Random Forest, Naive Bayes, DT, kNN, MLP	98.63
InSDN: SDN intrusion dataset [27]	Without feature selection	kNN, NB, Adaboost, DT, RF, rbf-SVM, lin-SVM, MLP	99
DDOS attack SDN Dataset [28]	Without feature selection	CNN, LSTM, CNN-LSTM, SVC-SOM-SAE-MLP	99.75
DDOS attack SDN Dataset	NCA	kNN, ANN, DT, SVM	100

The results show that machine learning models are quite successful in detecting attack traffic. Our work aims to contribute to the research conducted in this field. Our experimental results showed that using the NCA feature selection method on SDN traffic data increases the accuracy of machine learning methods in detecting attack traffics. While selecting features with the NCA algorithm, all features are scored according to their distinctiveness index values. However, although this feature selection method does not give the optimum number of features to be selected, it is a deficiency of the method, but experimental studies have been carried out by selecting a different number of features in this study. For attacks such as DDoS attacks that need to be intervened without wasting time, it is important to detect the attack traffic by using system resources as efficiently as possible. Therefore, the most effective features should be selected when creating machine learning models.

It can be seen from Table 8 that the performance of machine learning models in studies using feature selection algorithms is better than in other studies [10,19,21]. It can be said that model classification performance contributes positively to the classification of attack traffic when used in conforming to feature selection algorithms. However, given that

studies in the literature are run by applying different models on different datasets, it is difficult to make general evaluations on comparative results.

6. Conclusions

In this study, normal and attack traffic in the dataset obtained from the SDN environment was classified using machine learning algorithms. The customized SDN-based dataset consists of TCP, UDP, and ICMP normal and attack traffics. The dataset has statistical features such as byte_count, duration_sec, packet rate, and packet per flow except for features that define source and target machines. The NCA algorithm has been used to perform an effective classification and to select the most suitable features. After analyzing 22 network features NCA algorithms, 14 effective features were selected and given as input to machine learning algorithms. More than 100 thousand network records were classified by *k*NN, DT, ANN, and SVM algorithms after preprocessing and feature selection. The experimental results show that DT has a better accuracy rate than the other algorithms with 100%.

In future studies, it is planned to increase the diversity of attacks and compare the classification performances of machine learning models with feature selection algorithms.

Author Contributions: Conceptualization, Ö.T, and H.P.; methodology, Ö.T., Z.C. and E.B.; software, E.B. and Z.C.; validation, Ö.T., H.P., E.B. and Z.C.; formal analysis, Ö.T., H.P. and Z.C.; investigation, Ö.T., Z.C. and R.K.; resources, Ö.T., E.B., Z.C. and R.K.; data curation, E.B. and Z.C.; writing—original draft preparation, Ö.T. and Z.C.; writing—review and editing, Ö.T., H.P. and Z.C.; visualization, E.B. and Z.C.; supervision, Ö.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shu, Z.; Wan, J.; Li, D.; Lin, J.; Vasilakos, A.V.; Imran, M. Security in Software-Defined Networking: Threats and Countermeasures. *Mob. Netw. Appl.* **2016**, *21*, 764–776. [CrossRef]
2. Chica, J.C.C.; Imbachi, J.C.; Vega, J.F.B. Security in SDN: A comprehensive survey. *J. Netw. Comput. Appl.* **2020**, *159*, 102595. [CrossRef]
3. Nazih, W.; Elkilani, W.S.; Dhahri, H.; Abdelkader, T. Survey of countering DoS/DDoS attacks on SIP based VoIP networks. *Electronics* **2020**, *9*, 1827. [CrossRef]
4. Horak, T.; Strelec, P.; Huraj, L.; Tanuska, P.; Vaclavova, A.; Kebisek, M. The vulnerability of the production line using industrial IoT systems under ddos attack. *Electronics* **2021**, *10*, 381. [CrossRef]
5. Hu, C.; Han, L.; Yiu, S.M. Efficient and secure multi-functional searchable symmetric encryption schemes. *Secur. Commun. Netw.* **2016**, *9*, 34–42. [CrossRef]
6. Praseed, A.; Thilagam, P.S. DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 661–685. [CrossRef]
7. Mahjabin, T.; Xiao, Y.; Sun, G.; Jiang, W. A survey of distributed denial-of-service attack, prevention, and mitigation techniques. *Int. J. Distrib. Sens. Netw.* **2017**, *13*. [CrossRef]
8. Yusof, M.A.M.; Ali, F.H.M.; Darus, M.Y. Detection and Defense Algorithms of Different Types of DDoS Attacks. *Int. J. Eng. Technol.* **2018**, *9*, 410–444. [CrossRef]
9. Perez-Diaz, J.A.; Valdovinos, I.A.; Choo, K.K.R.; Zhu, D. A Flexible SDN-Based Architecture for Identifying and Mitigating Low-Rate DDoS Attacks Using Machine Learning. *IEEE Access* **2020**, *8*, 155859–155872. [CrossRef]
10. Sahoo, K.S.; Tripathy, B.K.; Naik, K.; Ramasubbareddy, S.; Balusamy, B.; Khari, M.; Burgos, D. An Evolutionary SVM Model for DDOS Attack Detection in Software Defined Networks. *IEEE Access* **2020**, *8*, 132502–132513. [CrossRef]
11. Kyaw, A.T.; Oo, M.Z.; Khin, C.S. Machine-Learning Based DDOS Attack Classifier in Software Defined Network. In Proceedings of the 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Phuket, Thailand, 24–27 June 2020; pp. 431–434. [CrossRef]
12. Janarthanam, S.; Prakash, N.; Shanthakumar, M. Adaptive Learning Method for DDoS Attacks on Software Defined Network Function Virtualization. *EAI Endorsed Trans. Cloud Syst.* **2020**, *6*, 166286. [CrossRef]
13. Tan, L.; Pan, Y.; Wu, J.; Zhou, J.; Jiang, H.; Deng, Y. A New Framework for DDoS Attack Detection and Defense in SDN Environment. *IEEE Access* **2020**, *8*, 161908–161919. [CrossRef]

14. Wang, L.; Liu, Y. A DDoS Attack Detection Method Based on Information Entropy and Deep Learning in SDN. In Proceedings of the IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 12–14 June 2020; pp. 1084–1088. [CrossRef]
15. Deepa, V.; Sudar, K.M.; Deepalakshmi, P. Design of Ensemble Learning Methods for DDoS Detection in SDN Environment. In Proceedings of the International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 30–31 March 2019. [CrossRef]
16. Karan, B.V.; Narayan, D.G.; Hiremath, P.S. Detection of DDoS Attacks in Software Defined Networks. In Proceedings of the 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 20–22 December 2018; pp. 265–270. [CrossRef]
17. Cil, A.E.; Yildiz, K.; Buldu, A. Detection of DDoS attacks with feed forward based deep neural network model. *Expert Syst. Appl.* **2021**, *169*, 114520. [CrossRef]
18. Nam, T.M.; Phong, P.H.; Khoa, T.D.; Huong, T.T.; Nam, P.N.; Thanh, N.H.; Thang, L.X.; Tuan, P.A.; Dung, L.Q.; Loi, V.D. Self-organizing map-based approaches in DDoS flooding detection using SDN. In Proceedings of the 2018 International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, 10–12 January 2018; pp. 249–254. [CrossRef]
19. Adhikary, K.; Bhushan, S.; Kumar, S.; Dutta, K. Decision Tree and Neural Network Based Hybrid Algorithm for Detecting and Preventing Ddos Attacks in VANETS. *Int. J. Innov. Technol. Explor. Eng.* **2020**, *9*, 669–675. [CrossRef]
20. Hosseini, S.; Azizi, M. The hybrid technique for DDoS detection with supervised learning algorithms. *Comput. Netw.* **2019**, *158*, 35–45. [CrossRef]
21. Ujjan, R.M.A.; Pervez, Z.; Dahal, K.; Bashir, A.K.; Mumtaz, R.; González, J. Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. *Futur. Gener. Comput. Syst.* **2020**, *111*, 763–779. [CrossRef]
22. Ravi, N.; Shalinie, S.M. Learning-Driven Detection and Mitigation of DDoS Attack in IoT via SDN-Cloud Architecture. *IEEE Internet Things J.* **2020**, *7*, 3559–3570. [CrossRef]
23. Yong, B.; Wei, W.; Li, K.C.; Shen, J.; Zhou, Q.; Wozniak, M.; Połap, D.; Damaševičius, R. Ensemble machine learning approaches for webshell detection in Internet of things environments. *Trans. Emerg. Telecommun. Technol.* **2020**, e4085. [CrossRef]
24. Kushwah, G.S.; Ranga, V. Optimized extreme learning machine for detecting DDoS attacks in cloud computing. *Comput. Secur.* **2021**, *105*, 102260. [CrossRef]
25. Damasevicius, R.; Venckauskas, A.; Grigaliunas, S.; Toldinas, J.; Morkevicius, N.; Aleliunas, T.; Smuikys, P. Litnet-2020: An annotated real-world network flow dataset for network intrusion detection. *Electronics* **2020**, *9*, 800. [CrossRef]
26. Erhan, D.; Anarım, E. Boğaziçi University distributed denial of service dataset. *Data Brief* **2020**, *32*, 106187. [CrossRef]
27. Elsayed, M.S.; Le-Khac, N.A.; Jurcut, A.D. InSDN: A novel SDN intrusion dataset. *IEEE Access* **2020**, *8*, 165263–165284. [CrossRef]
28. Ahuja, N.; Singal, G.; Mukhopadhyay, D. DLSDN: Deep learning for DDOS attack detection in software defined networking. In Proceedings of the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 28–29 January 2021; pp. 683–688. [CrossRef]
29. Ahuja, N.; Singal, G.; Mukhopadhyay, D. *"DDOS attack SDN Dataset", Mendeley Data, V1*; Bennett University: Greater Noida, India, 2020. [CrossRef]
30. Shao, E. Encoding IP Address as a Feature for Network Intrusion Detection. Ph.D. Dissertation, Purdue University Graduate School, West Lafayette, Indiana, 2019.
31. Fix, E.; Hodges, J.L. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *Int. Stat. Rev. Rev. Int. Stat.* **1989**, *57*, 238–247. [CrossRef]
32. Akbulut, Y.; Sengur, A.; Guo, Y.; Smarandache, F. NS-k-NN: Neutrosophic Set-Based k-Nearest Neighbors Classifier. *Symmetry* **2017**, *9*, 179. [CrossRef]
33. Altuntaş, Y.; Kocamaz, A.F.; Cömert, Z.; Cengiz, R.; Esmeray, M. Identification of Haploid Maize Seeds using Gray Level Co-occurrence Matrix and Machine Learning Techniques. In Proceedings of the 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 28–30 September 2018; pp. 1–5.
34. Cömert, Z. Fusing fine-tuned deep features for recognizing different tympanic membranes. *Biocybern. Biomed. Eng.* **2020**, *40*, 40–51. [CrossRef]
35. Cömert, Z.; Kocamaz, A.F. Comparison of Machine Learning Techniques for Fetal Heart Rate Classification. *Acta Phys. Pol. A* **2017**, *132*, 451–454. [CrossRef]
36. Hagan, M.T.; Demuth, H.B.; Beale, M.H.; De Jesús, O.; De Jesús, O. *Neural Network Design*, 2nd ed.; Hagan, M.T., Ed.; 2014. Available online: https://www.amazon.com/Neural-Network-Design-Martin-Hagan/dp/0971732116 (accessed on 21 May 2021) ISBN 9780971732117.
37. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning*; Springer: Berlin/Heidelberg, Germany, 2009; ISBN 9780387848570.
38. Diker, A.; Cömert, Z.; Avci, E.; Velappan, S. Intelligent system based on Genetic Algorithm and support vector machine for detection of myocardial infarction from ECG signals. In Proceedings of the 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, 2–5 May 2018; pp. 1–4.
39. Alkasassbeh, M. An empirical evaluation for the intrusion detection features based on machine learning and feature selection methods. *arXiv* **2017**, arXiv:1712.09623.

40. Frid-Adar, M.; Diamant, I.; Klang, E.; Amitai, M.; Goldberger, J.; Greenspan, H. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing* **2018**, *321*, 321–331. [CrossRef]
41. Raghu, S.; Sriraam, N. Classification of focal and non-focal EEG signals using neighborhood component analysis and machine learning algorithms. *Expert Syst. Appl.* **2018**, *113*, 18–32. [CrossRef]
42. Yang, W.; Wang, K.; Zuo, W. Neighborhood Component Feature Selection for High-Dimensional Data. *JCP* **2012**, *7*, 161–168. [CrossRef]
43. Başaran, E.; Cömert, Z.; Çelik, Y.; Budak, Ü.; Şengür, A. Otitis media diagnosis model for tympanic membrane images processed in two-stage processing blocks. *IOP Sci.* **2020**, *14*, 1–27.
44. Başaran, E.; Cömert, Z.; Çelik, Y.; Velappan, M.T.S. Determination of Tympanic Membrane Region in the Middle Ear Otoscope Images with Convolutional Neural Network Based YOLO Method. *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen Ve Mühendislik Derg.* **2020**, *2*, 919–928. [CrossRef]
45. Zhou, Q.; Ooka, R. Influence of data preprocessing on neural network performance for reproducing CFD simulations of non-isothermal indoor airflow distribution. *Energy Build.* **2021**, *230*, 110525. [CrossRef]
46. Rostami, M.; Berahmand, K.; Nasiri, E.; Forouzande, S. Review of swarm intelligence-based feature selection methods. *Eng. Appl. Artif. Intell.* **2021**, *100*, 104210. [CrossRef]