Article

Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks

Abdulsalam O. Alzahrani and Mohammed J. F. Alenazi *

Department of Computer Engineering, CCIS, King Saud University, Riyadh 11461, Saudi Arabia; 439105796@student.ksu.edu.sa
* Correspondence: mjalenazi@ksu.edu.sa

Abstract: Software-defined Networking (SDN) has recently developed and been put forward as a promising and encouraging solution for future internet architecture. Managed, the centralized and controlled network has become more flexible and visible using SDN. On the other hand, these advantages bring us a more vulnerable environment and dangerous threats, causing network breakdowns, systems paralysis, online banking frauds and robberies. These issues have a significantly destructive impact on organizations, companies or even economies. Accuracy, high performance and real-time systems are essential to achieve this goal successfully. Extending intelligent machine learning algorithms in a network intrusion detection system (NIDS) through a software-defined network (SDN) has attracted considerable attention in the last decade. Big data availability, the diversity of data analysis techniques, and the massive improvement in the machine learning algorithms enable the building of an effective, reliable and dependable system for detecting different types of attacks that frequently target networks. This study demonstrates the use of machine learning algorithms for traffic monitoring to detect malicious behavior in the network as part of NIDS in the SDN controller. Different classical and advanced tree-based machine learning techniques, Decision Tree, Random Forest and XGBoost are chosen to demonstrate attack detection. The NSL-KDD dataset is used for training and testing the proposed methods; it is considered a benchmarking dataset for several state-of-the-art approaches in NIDS. Several advanced preprocessing techniques are performed on the dataset in order to extract the best form of the data, which produces outstanding results compared to other systems. Using just five out of 41 features of NSL-KDD, a multi-class classification task is conducted by detecting whether there is an attack and classifying the type of attack (DDoS, PROBE, R2L, and U2R), accomplishing an accuracy of 95.95%.

Keywords: network intrusion detection system (NIDS); software defined network (SDN); NSL-KDD; XGBoost; machine learning; feature scaling

check for updates

Citation: Alzahrani, A.O.; Alenazi, M.J.F. Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks. *Future Internet* **2021**, *13*, 111. https://doi.org/10.3390/fi13050111

Academic Editor: Rajan Shankaran

Received: 20 March 2021
Accepted: 27 April 2021
Published: 28 April 2021

1. Introduction

A network intrusion detection system is a process for discovering the existence of malicious or unwanted packets in the network. This process is done using real-time traffic monitoring to find out if any unusual behavior is present in the network or not. Big data, powerful computation facilities, and the expansion of the network size increase the demand for the required tasks that should be carried out simultaneously in real-time. Therefore, NIDS should be careful, accurate, and precise in monitoring, which has not been the case in the traditional methods. On the other hand, the rapid increase in the accuracy of machine learning algorithms is highly impressive. Its introduction relies on the increasing demand for improved performance on different types of network [1]. However, software defined network (SDN) implementation of the network-based intrusion detection system (NIDS) has opened a frontier for its deployment, considering the increasing scope and typology of security risks of modern networks. The rapid growth in the volume of network data and connected devices carries inherent security risks. The adoption of technologies such as the Internet of Things (IoT), artificial intelligence (AI), and quantum computing, has increased the threat level [2], making network security challenging and necessitating a new paradigm in its implementation. Various attacks have overwhelmed previous approaches (classified into signature-based intrusion detection systems [3,4] and anomaly-based intrusion detection systems [5,6]), increasing the need for advanced, adaptable and resilient security implementation [1,2,7,8]. For this reason, the traditional network design platform is being transformed into the evolving SDN implementation [9]. Monitoring data and analyzing it over time are essential to the process of predicting future events, such as risks, attacks and diseases. The more details are formed, discovered and documented through analyzing very large-scale data, the more saved resources, as well as the working environment, will remain normal without any variations. Big data analytics (BDA) research in the supply chain becomes the secret of a protector for managing and preventing risks [10]. BDA for humanitarian supply chains can aid the donors in their decision of what is appropriate in situations such as disasters, where it can improve the response and minimize human suffering and deaths. BDA and data monitoring using machine learning can help in identifying and understanding the interrelationships between the reasons, difficulties, obstacles and barriers that guide organizations in taking the most efficient and accurate decisions in risk management processes. This could impact entire organizations and countries, producing a hugely significant improvement in the process.

Network monitoring-based machine learning techniques have been involved in diverse fields. Using bi-directional long-short-term-memory neural networks, a social media network monitoring system is proposed for analyzing and detecting traffic accidents [11]. The proposed method retrieves traffic-related information from social media (Facebook and Twitter) using query-based crawling: this process collects sentences related to any traffic events, such as jams, road closures, etc. Subsequently, several pre-processing techniques are carried out, such as steaming, tokenization, POS tagging and segmentation, in order to transform the retrieved data into structured form. Thereafter, the data are automatically labeled as ’traffic’ or ’non-traffic’, using a latent Dirichlet allocation (LDA) algorithm. Traffic- labeled data are analyzed into three types; positive, negative, and neutral. The output from this stage is a sentence labeled according to whether it is traffic or non-traffic, and with the polarity of that traffic sentence (positive, negative or neutral). Then, using the bag-of-words (BoW) technique, each sentence is transformed into a one-hot encoding representation in order to feed it to the Bi-directional LSTM neural network (Bi-LSTM). After the learning process, the neural networks perform multi-class classification using the softmax layer in order to classify the sentence in terms of location, traffic event and polarity types. The proposed method compares different classical machine learning and advanced deep learning approaches in terms of accuracy, F-score and other criteria.

Many initiatives and workshops have been conducted in order to improve and develop the healthcare systems using machine learning, such as [12]. In these workshops several proposed machine algorithms have been used, such as K Nearest-Neighbors, logistic regression, K-means clustering, Random Forest (RF) etc, together with deep learning algorithms such as CNN, RNN, fully connected layer and auto-encoder. These varieties of techniques allow the researchers to deal with several data types, such as medical imaging, history, medical notes, video data, etc. Therefore, different topics and applications are introduced, with significant performance results such as causal inference, in investigations of Covid-19, disease prediction, such as disorders and heart diseases.

Using intelligent ensemble deep learning methods, healthcare monitoring is carried out for prediction of heart diseases [13]. Real-time health status monitoring can prevent and predict any heart attacks before occurrence. For disease prediction, the proposed ensemble deep learning approach achieved a brilliant accuracy performance score of 98.5%. The proposed model takes two types of data that are transferred and saved on an online cloud database. The first is the data transferred from the sensors; these sensors have been placed in different places on the body in order to extract more than 10 different types of medical data. The second type is the daily electronic medical records from doctors, which includes various types of data, such as smoking history, family diseases, etc. The features are fused using the feature fusion Framingham Risk factors technique, which executes two tasks at a time, fusing the data together, and then extracting a fused and informative feature from this data. Then different pre-processing techniques are used to transform the data into a structured and well-prepared form, such as normalization, missing values filtering and feature weighting. Subsequently, an ensemble deep learning algorithm starts which learns from the data in order to predict whether a heart disease will occur or the threat is absent.

IDS refers to a mechanism capable of identifying or detecting intrusive activities. In a broader view, this encompasses all the processes used in the discovery of unauthorized uses of network devices or computers. This is achieved through software designed specifically to detect unusual or abnormal activities. IDS can be classified according to several surveys and sources in the literature [14–17] into four types (HIDS, NIDS, WIDS, NBA).

NIDS is an inline or passive-based intrusion detection technique. The scope of its detection targets network and host levels. The only architecture that fits and works with NIDS is the managed network. The advantage of using NIDS is that it costs less and is quicker in response, since there is no need to maintain sensor programming at the host level. The performance of monitoring the traffic is close to real-time; NIDS can detect attacks as they occur. However, it has the following limited features. It does not indicate if such attacks are successful or not: it has restricted visibility inside the host machine. There is also no effective way to analyze encrypted network traffic to detect the type of attack [16,18]. Moreover, NIDS may have difficulty capturing all packets in a large or busy network. Thus, it may fail to recognize an attack launched during a period of high traffic.

SDN provides a novel means of network implementation, stimulating the development of a new type of network security application [19]. It adopts the concept of programmable networks through the deployment of logically centralized management. The network deployment and configuration are virtualized to simplify complex processes, such as orchestration, network optimization, and traffic engineering [20–22]. It creates a scalable architecture that allows sufficient and reliable services based on certain types of traffic [23]. The global view approach to a network enhances flow-level control of the underlying layers.

Implementing NIDS over SDN becomes a major effective security defense mechanism for detecting network attacks from the network entry point [24]. NIDS has been implemented and investigated for decades to achieve optimal efficiency. It represents an application or device for monitoring network traffic for suspicious or malicious activity with policy violations [25]. Such activities include malware attacks, untrustworthy users, security breaches, and DDoS. NIDS focuses on identifying anomalous network traffic or behavior; its efficiency means that network anomaly is adequately implemented as part of the security implementation. Since it is nearly impossible to prevent threats and attacks, NIDS will ensure early detection and mitigation [26]. However, the advancement in NIDS has not instilled sufficient confidence among practitioners, since most solutions still use less capable, signature-based techniques [2].

This study aims to increase the focus on several points:

- choosing the right algorithm for the right tasks depends on the data types, size and network behavior and needs.
- Implementing the optimized development process by preparing and selecting the benchmark dataset in order to build a promising system in NIDS.
- Analyzing the data, finding, shaping, and engineering the important features, using several preprocessing techniques by stacking them together with an intelligent order to find the best accuracy with the lowest amount of data representation and size.
- proposing an integration and complete development process using those algorithms and techniques from the selection of dataset to the evaluation of the algorithms using a different metric. Which can be extended to other NIDS applications.

This study enhances the implementation of NIDS by deploying different machine learning algorithms over SDN. Tree-based machine learning algorithms (XGBoost, random forest (RF), and decision tree (DT)) were implemented to enhance the monitoring and accuracy performance of NIDS. The proposed method will be trained on network traffic packet data, collected from large-scale resources and servers called NSL-KDD dataset to perform two tasks at a time by detecting whether there is an attack or not and classifying the type of attack.

This study enhances the implementation of NIDS by deploying machine learning over SDN. Tree-based machine learning algorithms (XGBoost, random forest (RF), and decision tree (DT)) are proposed to enhance NIDS. The proposed method will be trained on network traffic packet data, collected from large-scale resources and servers, called the NSL-KDD dataset to perform two tasks at a time by detecting whether there is an attack or not and classifying the type of attack.

The remainder of the paper is organized as follows. Section 2 presents the background and related work. In Section 3, materials and methods are introduced and illustrated. Section 5 discusses and evaluates the results with multiple evaluation metrics. Finally, Section 6 presents the conclusion and recommendations for future study.

2. Background and Related Work

Integrating machine learning algorithms into SDN has attracted significant attention. In [26], a solution was proposed that solved the issues in KDD Cup 99 by performing an extensive experimental study, using the NSL-KDD dataset to achieve the best accuracy in intrusion detection. The experimental study was conducted on five popular and efficient machine learning algorithms (RF, J48, SVM, CART, and Naïve Bayes). The correlation feature selection algorithm was used to reduce the complexity of features, resulting in 13 features only in the NSL-KDD dataset.

This study tests the NSL-KDD dataset's performance for real-world anomaly detection in network behavior. Five classic machine learning models RF, J48, SVM, CART, and Naïve Bayes were trained on all 41 features against the five normal types of attacks, DOS, probe, U2R, and R2L to achieve average accuracies of 97.7%, 83%, 94%, 85%, and 70% for each algorithm, respectively. The same models were trained again using the reduced 13 features to achieve average accuracies of 98%, 85%, 95%, 86%, and 73% for each model. In [27], a deep neural network model was proposed to find and detect intrusions in the SDN. The NSL-KDD dataset was used to train and test the model. The neural network was constructed with five primary layers, one input layer with six inputs, three hidden layers with (12, 6, 3) neurons, and one output layer with 2D dimensions. The proposed method was trained on six features chosen from 41 features in the NSL-KDD dataset, which are basic and traffic features that can easily be obtained from the SDN environment. The proposed method calculates the accuracy, precision and recall, achieving an F1-score of 0.75. A second evaluation was conducted on seven classic machine learning models (RF, NB, NB Tree, J48, DT, MLP, and SVM) proposed in [28] and the model achieved sixth place out of eight. The same author [29] extended the approach using a gated recurrent unit neural network (GRU-RNN) for SDN anomaly detection, achieving accuracy up to 89%. In addition, the Min-Max normalization technique is used for feature scaling to improve and boost the learning process.

The SVM classifier [30], integrated with the principal component analysis (PCA) algorithm, was used for an intrusion detection application. The NSL-KDD dataset is used in this approach to train and optimize the model for detecting abnormal patterns. A Min-Max normalization technique was proposed to solve the diversity data scale ranges with the lowest misclassification errors [31]. The PCA algorithm is selected as a statistical technique to reduce the NSL-KDD dataset's complexity, reducing the number of trainable parameters that needed to be learned. The nonlinear radial basis function kernel was chosen for SVM optimization. Detection rate (DR), false alarm rate (FAR), and correlation coefficient metrics were chosen to evaluate the proposed model, with an overall average accuracy of 95% using 31 features in the dataset. In [32], an extreme gradient-boosting (XGBoost) classifier was used to distinguish between two attacks, i.e., normal and DoS. The detection method was analyzed and conducted over POX SDN, as a controller, which is an SDN open-source platform for prototyping and developing a technique based on SDN. Mininet was used to emulate the network topology to simulate real-time SDN-based cloud detection. Logistic regression was selected as a learning algorithm, with a regularization term penalty to prevent overfitting. The XGBoost term was added and combined with the logistic regression algorithm to boost the computations by constructing structure trees. The dataset used in this approach was KDD Cup 1999, while 400 K samples were selected for constructing the training set. Two types of normalization techniques were used; one with a logarithmic-based technique and one with a Min-Max-based technique. The average overall accuracy for XGBoost, compared to RF and SVM, was 98%, 96%, 97% respectively.

Based on DDoS attack characteristics, a detection system was simulated with the Mininet and floodlight platform using the SVM algorithm [5]. The proposed method categorizes the characteristics into six tuples, which are calculated from the packet network. These characteristics are the speed of the source IP (SSIP), the speed of the source port, the standard deviation of flow packets, the deviation of flow bytes (SDFB), the speed of flow entries, and the ratio of pair-flow. Based on the calculated statistics from the SVM classifier's six characteristics, the current network state is normal or attack. Attack flow (AF), DR, and FAR were chosen to achieve an average accuracy of 95%.

In TSDL [33] a model with two stages of deep neural networks was designed and proposed for NIDS, using a stacked auto-encoder, integrated with softmax in the output layer as a classifier. TSDL was designed and implemented for Multi-class classification of attack detection. Down-sampling and other preprocessing techniques were performed over different datasets in order to improve the detection rate, as well as the monitoring efficiency. The detection accuracy for UNSW-NB15 was 89.134%. Different models of neural networks, such as variational auto-encoder, seq2seq structures using Long-Short-Term-Memory (LSTM) and fully connected networks were proposed in [34] for NIDS. The proposed approach was designed and implemented to differentiate between normal and attack packets in the network, using several datasets, such as NSL-KDD, UNSW-NB15, KYOTO-HONEYPOT, and MAWILAB. A variety of preprocessing techniques have been used, such as one-hot-encoding, normalization, etc., for data preparation, feature manipulation and selection and smooth training in neural networks. Those factors are designed mainly, but not only, to enable the neural networks to learn complex features from different scopes of a single packet. Using 4 hidden layers, a deep neural network model [35] was illustrated and implemented on KDD cup99 for monitoring intrusion attacks. Feature scaling and encoding were used for data preprocessing and lower data usage. More than 50 features were used to perform this task on different datasets. Therefore, complex hardware GPUs were used in order to handle this huge number of features with lower training time.

A supervised [36] adversarial auto-encoder neural network was proposed for NIDS. It combined GANS and a variational auto-encoder. GANS consists of two different neural networks competing with each other, known as the generator and the discriminator. The result of the competition is to minimize the objective function as much as possible, using the Jensen-Shannon minimization algorithm. The generator tries to generate fake data packets, while the discriminator determined whether this data is real or fake; in other words, it checks if that packet is an attack or normal. In addition, the proposed method integrates the regularization penalty with the model structure for overfitting control behavior. The results were reasonable in the detection rate of U2RL and R2L but lower in others.

Multi-channel deep learning of features for NIDS was presented in [37], using AE involving CNN, two fully connected layers and the output to the softmax classifier. The evaluation is done over three different datasets; KDD cup99, UNSW-NB15 and CICIDS, with an average accuracy of 94%. The proposed model provides effective results; however, the structure and the characteristics of the attack were not highlighted clearly.

The proposed method enhances the implementation of NIDS by deploying machine learning over SDN. It introduces a machine learning algorithm for network monitoring within the NIDS implementation on the central controller of the SDN. In this paper, enhanced tree-based machine learning algorithms are proposed for anomaly detection. Using only five features, a multi-class classification task is conducted by detecting whether there is an attack or not and classifying the type of attack.

3. Proposed Method

In this section, we discuss and explain each component and its role in the NIDS architecture. As shown in Figure 1, the SDN architecture can be divided into three main layers, as follows:

3.1. System Architecture Layers

NIDS component architecture is constructed in three main parts as follows:

- The infrastructure layer consists of two main parts: hardware and software components. The hardware components are devices such as routers and switches. The software components are those components that interface with the hardware, such as OpenFlow switches.
- The control layer is an intelligent network controller, such as an SDN controller. The control layer is the layer responsible for regulating actions and traffic data management by establishing or denying every network flow.
- The application layer is the one that performs all network management tasks. These tasks can be performed using an SDN controller and NIDS.

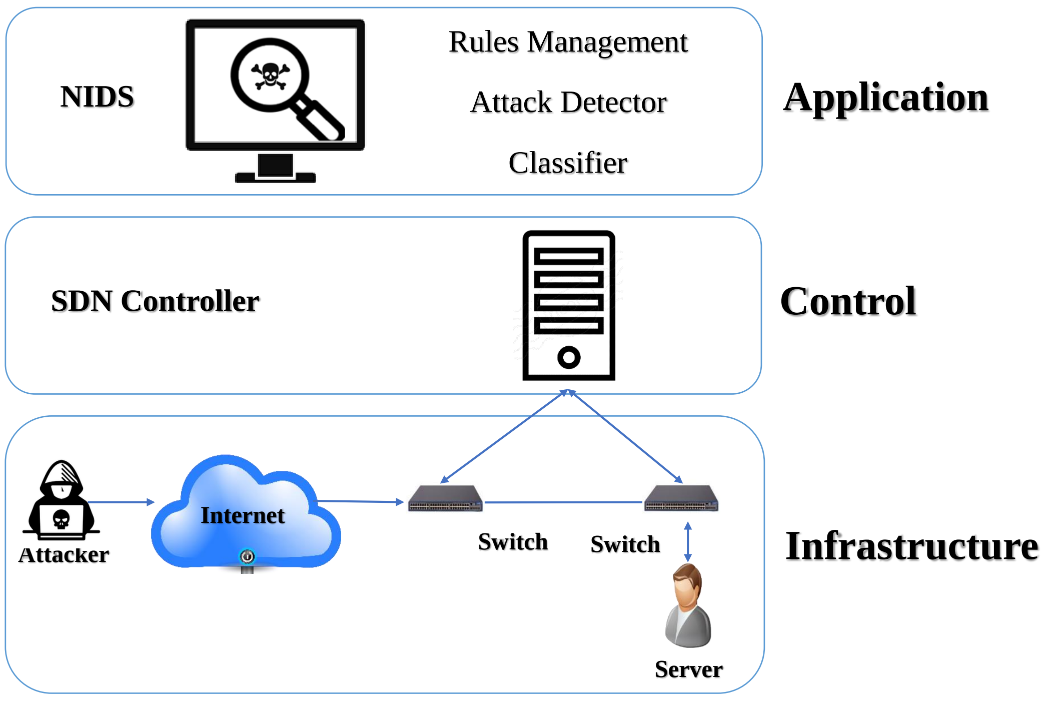

Figure 1. Proposed System Architecture consists of three main layers: the application layer, which contains the Rule management system and NIDS; the control layer contains the SDN controller, while the infrastructure layer includes the switches, servers and the attacker that may exist in one of those switches.

3.2. Proposed NIDS Scenario

Attacks are created by an attacker and delivered through the internet. NIDS is deployed over the SDN controller. As NIDS listens to the network and actively compares all traffic against predefined attack signatures, it detects the attacker's scanning attempts. It sends an alert to administrators through its control, and the connections will be blocked due to specific rules in the firewall or routers.

4. Evaluation

This section presents a generalized flowchart of the proposed method. The dataset, pre-processing techniques, and proposed machine learning algorithms will be presented and discussed.

4.1. Generalized Block Diagram

In this subsection, a generalized block diagram is presented and discussed. As shown in Figure 2, the NSL-KDD dataset is used. Data analysis, feature engineering, and other pre-processing techniques are conducted to train the model, using the best hyperparameters, with only five features. Tree-based algorithms are used for the multi-class classification task. The processed data enter the algorithm and are classified as to whether they constitute an attack or are normal; then, the type of attack will be analyzed to see which category it belongs to, and action is taken accordingly.

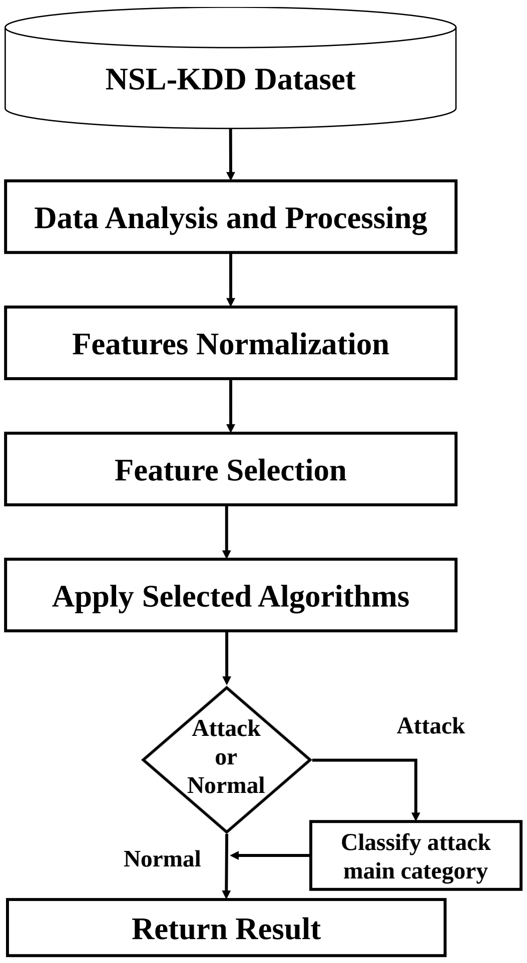

Figure 2. Proposed System Architecture. This flow chart represents the overall process. The process starts with the data source, the NSL-KDD dataset, followed by data analysis and some pre-processing techniques for data cleaning. Feature normalization and feature selection are performed in order to implement the later selected tree-based machine learning algorithm to classify whether there is an attack or not and the types of attack.

4.2. Dataset Overview

The KDD Cup is the leading data mining competition in the world. The NSL-KDD dataset was proposed to solve many issues represented in the KDD Cup 1999 dataset. Many researchers have used the NSL-KDD dataset to develop and evaluate the NIDS problem. The dataset includes all types of attacks. The dataset has 41 features, categorized into three main types (basic feature, content-based, and traffic-based features) and labeled as either normal or attack, with the attack type precisely categorized. The categories can be classified into four main groups Table 1, with a brief description of each attack type and its impact.

Table 1. Attacks in the NSL-KDD dataset and their impact.

Attack Name	Attack Type
Denial of Service (DoS)	Memory resources
User to Root (U2R)	Passwords, root access
Remote to Local (R2L)	Network resources
Probe	Network sniffing and security control

The authors propose the NSL-KDD dataset with two separate datasets; the training and test sets, with randomized sampling containing numeric and symbolic data. The training set is not at the same probability distribution as the test set, making the task more realistic to a real-world application problem. The NSL-KDD dataset does not have a redundant record in the training and test sets, which may cause the specific algorithm to be biased toward a specific record during the training (learning) process. As shown in Table 2, the generated dataset consists of training and test sets, each associated with 41 features. In the NSL-KDD dataset, there are no duplicated records in the test set, helping the algorithms during learning to have a better detection rate. The categories in the test set are excluded in the training, which is also advantageous for our selection, as stated in Tables 2 and 3.

Table 2. Training set and test set size.

Name	Training Set	Test Set
Number of samples	125,973	22,554

Table 3. Attack types in the NSL-KDD dataset and their descriptions.

Attack Categories	Training Set Attack Names	Test Set Attack Names
Dos	back, land, Neptune, pod, smurf, teardrop	back, land, neptune, pod, smurf, teardrop,(mailbomb), process table, udpstorm, apache2, worm
Probe	ipsweep, nmap, portsweep, satan	ipsweep, nmap, portsweep, satan, mscan, saint
U2R	Buffer overflow, load module, perl, rootkit	Buffer overflow, load module, perl, rootkit, sqlattack, xterm, pst
R2L	ftp-write, guess-passwd, imap, multihop, phf, spy, warezmaster.	ftp-write, guess-passwd, imap, multihop, phf, spy, warezmaster, xlock, xsnoop, snmpguess, snmpgetattack, HTTP tunnel, send-mail, named, warez client

As shown in Figures 3 and 4, dataset statistics are proposed for five attack categories for the training and test sets.

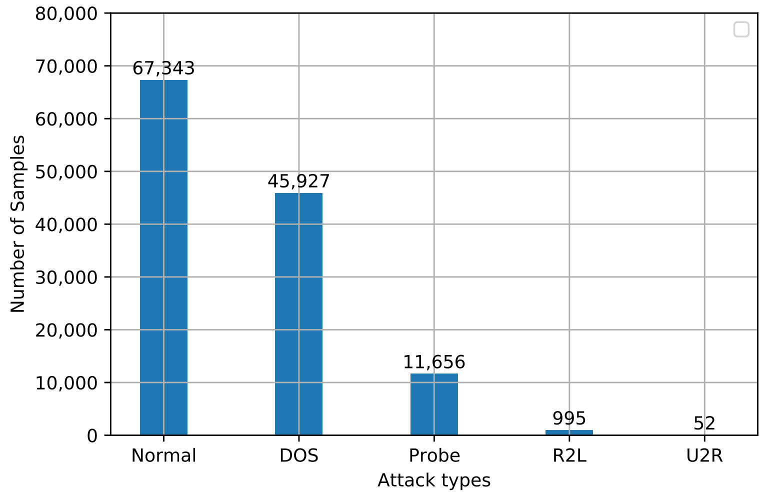

Figure 3. Categorized training set statistics show the number of samples in the four types of attacks and the normal samples in the training set.

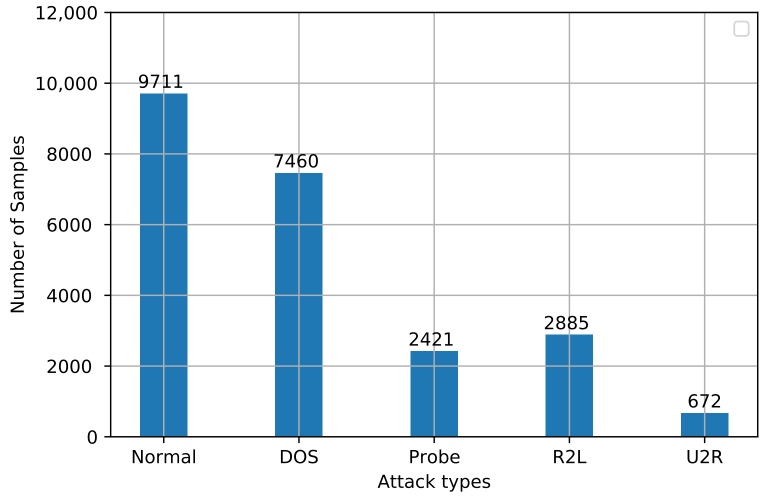

Figure 4. Categorized test set statistics show the number of samples in the four types of attacks and the normal samples in the test set. These samples are an unseen set, for model evaluation.

4.3. Training Features

As stated in the previous subsection, the dataset has 41 features labeled as either normal or attack with the precise attack category. After experimental trials, five features were selected out of the 41 features in the NSL-KDD dataset, which have the most impact and effect on algorithm-learning performance. Table 4 presents the selected five features with a brief description.

Table 4. Selected features with their description.

Feature Name	Description
Duration	length (number of seconds) of the connection.
protocol-type	type of protocol, such as tcp, udp, icmp
src-bytes	number of data bytes from source to destination
srv-count	number of connections to the same service as the current connection in the past two seconds
dst-host-same-src-port-rate	percentage of connections from the port services to the destination host.

4.4. Pre-Processing Techniques (Normalization)

The features dataset contains values that have different scale ranges during learning and minimize the loss function. These scales affect each iteration step during the gradient optimization process, which tends to affect the learning rate optimization, since we want the model to converge to the global or local minimum quickly and accurately. Min-Max normalization has several advantages over the standard scaling methods. Min-Max scaling has the ability to deal with the distribution of features that are not Gaussian, since the anomaly detection applications have no specific distribution to follow, unlike the signature-based method in NIDS, which is very suitable in our NSL-KDD dataset. The Min-Max normalization technique is proposed to prevent the gradient while optimizing the loss function from the un-smoothing path toward the global minimum. It takes the column's minimum value and maximum value, where the output values range between 0 and 1, as shown in the following equation.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

where $X - min$ is the smallest number in the column, $X - max$ is the largest number in the column, and X is the original data sample value to be normalized.

4.5. Baseline Algorithms

This section presents the details of the baseline algorithms used for NIDS; namely, DT, XGBoost, and RF.

- Decision Tree (DT)

The DT algorithm is a supervised tree-based method. It can be used for regression or classification tasks. DT creates and trains the model to predict the target's class value by learning simple tree rules. It starts from the first or the root node in the tree graph, compares the nodes' values with the given input, and searches in depth until it finds the required answer. DT can be constructed from the root node, which represents the entire record. The splitting process, which is the technique that makes the DT, works with it. The third decision node represents the node that has been chosen by the model after one iteration selection — the fourth terminal node, which is the last level in the tree or the sub-tree. The ID3 metric is used to build the tree. This algorithm uses the entropy function and information gain as a metric, as shown in the following equation.

$$H(S) = \sum_{c \in C} (-P(c) \log_2 P(c)) \tag{2}$$

Here, S is the current dataset for which the entropy is being calculated. C is the set of classes in S, $C = Yes, No$. $P(c)$ is the proportion of the number of elements in class c to the number of elements in set S. Using the ID3 and entropy metrics, the attributes are chosen to split the decision node. The smallest entropy is the first candidate to be chosen and split.

- XGBoost

The XGBoost [38] algorithm is a tree-based method. It uses a gradient-boosting framework. It can be used in regression and classification-ranking tasks on a tabular or structured dataset, which fits our research. Using the depth-first search approach, XGBoost builds the trees and optimizes the gradient, using parallel processing criteria. It uses a regularization term penalty to avoid overfitting/bias phenomena during the training process. For a given dataset with n examples and m features (x,y), a tree ensemble model uses K additive functions to predict the output, as follows.

$$Y = \sum_{k=1}^{k} f_k(x_i), f_k \in F \tag{3}$$

Here, F is the space assigned to regression trees. XGBoost tries to optimize and minimize the loss function using the following regularized objective.

$$I() = \sum_{i}^{k} I(Y_{true}, Y_{predicted}) + \sum_{i}^{k} \Omega(f_k) \tag{4}$$

$$\Omega(f_k) = \gamma * T + \frac{1}{2}\lambda|W|^2 \tag{5}$$

where $\gamma * T$, is the complexity penalized term of the model and $\frac{1}{2}\lambda|W|^2$ is the regularization term penalty.

- Random Forest (RF)

RF is a supervised ensemble machine learning algorithm. The ensemble approach is used when multiple machine learning algorithms are used and combined. RF is constructed from multiple decision trees, which may achieve more accurate and stable results. RF is trained using bagging methods; it can be used in classification and regression tasks.

5. Results and Discussion

To evaluate the performance of NIDS in terms of accuracy (AC), different metrics were used; precision (P), recall (R), and F-measure (F). These metrics can be calculated using confusion matrix parameters: true positive (the number of anomalous instances that are correctly classified); false positive (the number of normal instances that are incorrectly classified as anomalous); true negative (the number of normal instances that are correctly classified); and false negative (the number of anomalous instances that are incorrectly classified as normal). A good NIDS must achieve high DR and FAR.

Accuracy (AC): This is the percentage of correctly classified network activities. As shown in Table 5.

Precision (P): The percentage of predicted anomalous instances that are actual anomalous instances; the higher P, the lower FAR. As shown in Table 5.

Recall (R): the percentage of predicted attack instances versus all attack instances presented. As shown in Table 5.

F-measure (F): measuring the performance of NIDS using the harmonic mean of the P and R. We aim to achieve a high F-score. As shown in Table 5.

Table 5. Detailed equations for different performance evaluation metrics.

Evaluation Metric Name	Equation
Accuracy (A)	(TP + FP)/(FP + TP + FN + TN)
Percision (P)	TP/(FP + TP)
Recall (R)	TP/(TP + FN)
F-score (F)	(2*P*R)/(P + R)

We compare XGBoost against the other two tree- based methods, RF and DT. Using the test set, which includes the four types of attacks as discussed in Figure 5. Three different evaluation metrics are computed; F-score, precision and recall. As shown in Figure 5, XGBoost ranked first in the evaluation, with an F1-score of 95.55% , while RF and DT achieved 94.6% and 94.5%, respectively. For precision, XGBoost outperformed RF and DT with a score of 92%, while RF and DT scored 90% and 90.2% respectively. Finally, for Recall, our proposed method with XGBoost proves its stability with a score of 98% while for RF and DT, the results were 82%, and 85%, respectively.

From these results, the proposed model with XGBoost performs with high precision and high recall, which means that the classifier returns accurate results and high precision, while, at the same time, returning a majority of all positive results (it's an attack and the classifier detects that it's an attack), which means high recall.

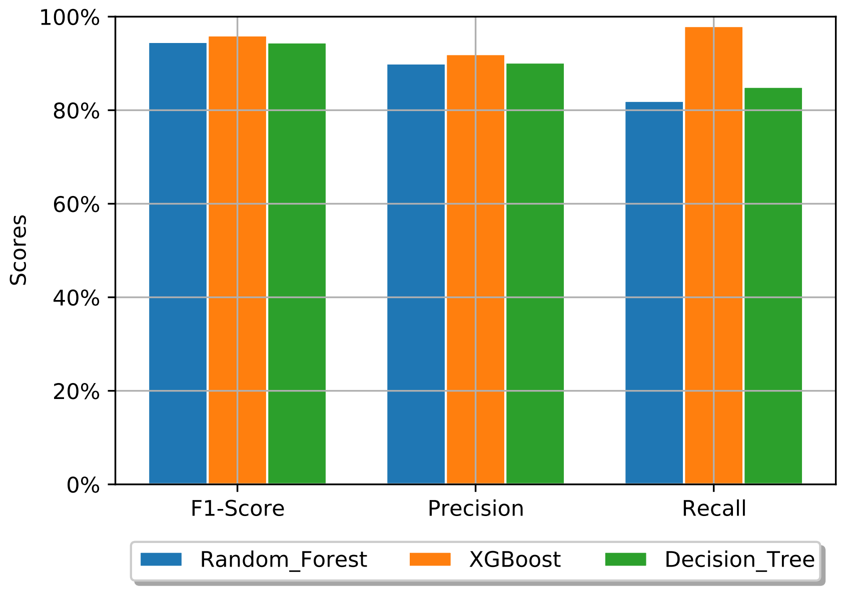

Figure 5. The proposed model XGBoost is compared with other tree-based machine learning algorithms, using F1-score, precision and recall as evaluation metrics.

In this evaluation, the accuracy metric is chosen in order to measure the performance of the proposed method against the results of Tang [27] on the binary classification task. The binary classification task represents the model's ability to detect whether there is an attack or not in the network environment. The accuracy metric measures tolerance, or how close the predicted classification values are to the true values. As shown in Figure 6, the proposed XGBoost algorithm with the selected five features achieved a high accuracy score of 95.55%, compared with the deep neural network with six features proposed by Tang [27], which had a score of 75.75%. In addition, from the previous Figure 5, we showed that the proposed model performs with high precision, with a score of 92%. Higher precision is related to reproducibility and repeatability; it is the degree to which repeated measurements on the classification task, under unchanged conditions, show the same results. We conclude from the previous scores that the proposed algorithm is accurate and stable in the binary classification task.

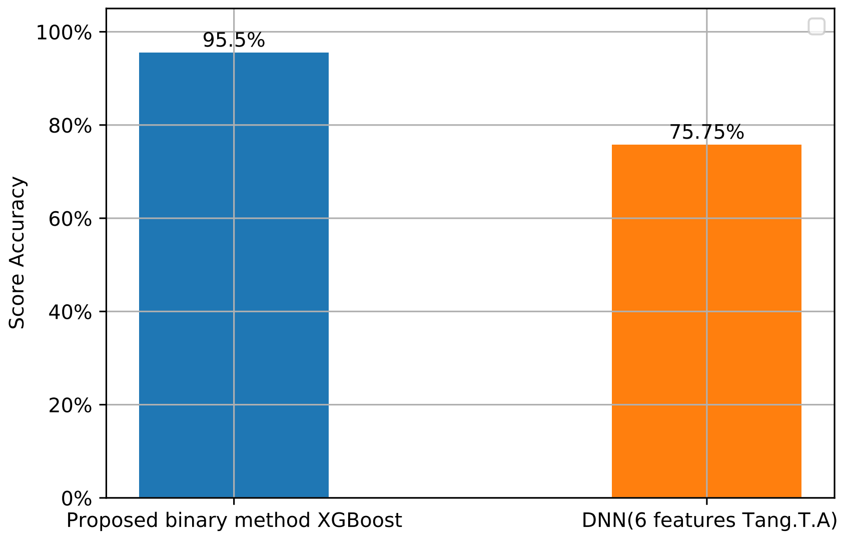

Figure 6. Accuracy score is measured in order to evaluate the performance of the proposed XGBoost model against the Deep Neural Networks model.

Deep inside the performance details of the proposed model, the performance of the multi-class classification against the different types of attacks is shown in Figure 7. F1-score, precision and recall are computed on the four attack categories separately. For the F1-score, the proposed XGBoost showed an outstanding performance in detecting R2L and U2R attacks with similar scores of 96%, while against DOS and Probe, the model achieved scores of 94.6% and 94.5%, respectively, with a difference of 2%.

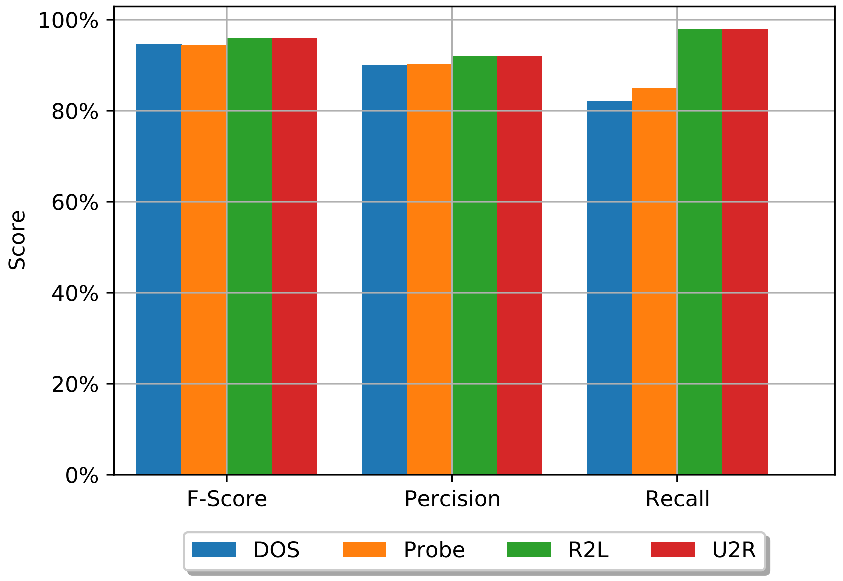

Figure 7. Performance evaluation of the proposed model in detecting attacks for each category, using F-score, precision and recall.

For precision and recall, the detection scores for R2L and U2RL achieved similar scores of 92% and 98%, respectively. For DOS and Probe, the model achieved slightly similar precision scores of 90%, 90.1%, and recall scores of 82% and 85%, respectively. We conclude from these F1-scores that the proposed model has a difference of only 2% in its performance against the four categories of attacks, while precision and recall differ by nearly 7%. This shows that the proposed model is forceful, effective and energetic, and can be applied to several types of attacks without weakening its performance.

For another evaluation of the proposed model on the four types of attacks, to make sure that the proposed method's performance is efficient and capable, we used Receiver Operating Characteristic (ROC), which plots and measures the trade-off between TPR (recall, or probability of detection), and FPR, known as the probability of false alarm. Afterwards, we compute the Area Under Curve or AUC, to measure the overall 2-D area under the all-ROC curve. AUC gives us the accumulated performance over all the possible classification thresholds. AUC is scale-invariant and threshold-invariant: it is not a function of threshold, like F-score; it is an evaluation of the classifier as threshold versus overall possible values. As shown in Figure 8 , the AUC value achieves 99.94% for all four classes. Hence, another evaluation method proves that the proposed method is qualified and skillful in attack classification.

In the fifth evaluation (Figure 9), we test the proposed method's detection rate against the binary approach proposed in [27]. The detection rate, or sensitivity, or true-positive rate, measures the proportion of the model's predictions of whether a packet is an attack or not with respect to all positive data points or all attacks. For instance, if we have 100 samples of packets, of which 95 are attacks, a robust model shows that it can detect and classify all the 95 attacks correctly and classify the other 5 as normal. As shown in Figure 9, the proposed method successfully accomplished an accuracy of 96.55% in the detection rate, while the deep learning method with six features achieved a score of 75.75%.

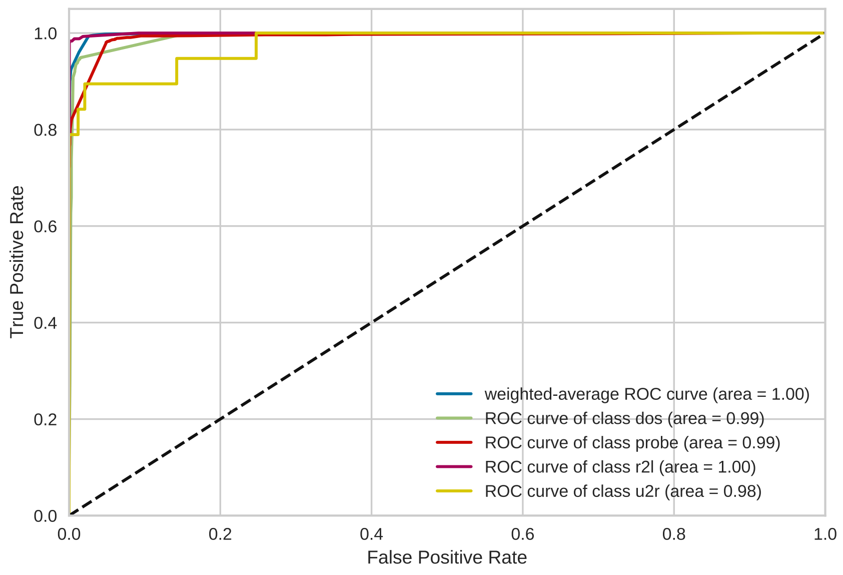

Figure 8. Computing the ROC curve by measuring the true-positive rate against the false-positive rate of the proposed model on each attack category (DOS, Probe, R2L, and U2R).

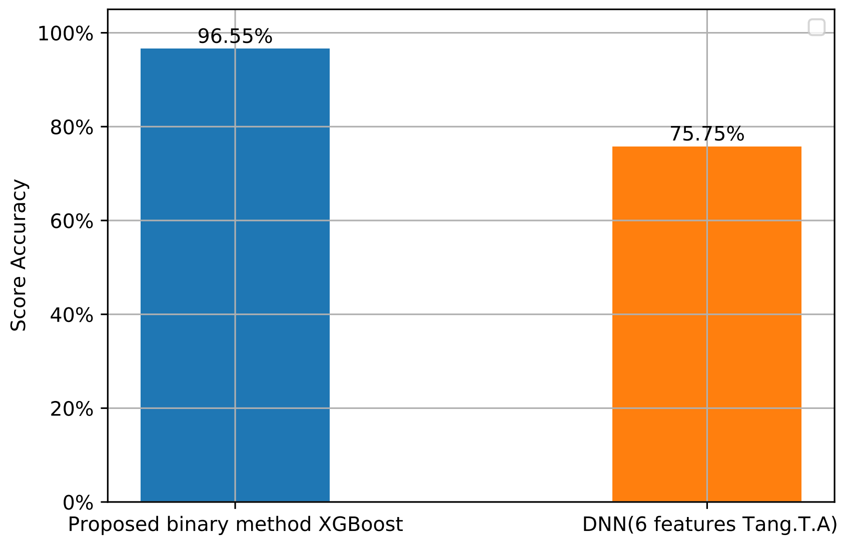

Figure 9. Attack detection rate comparison between the proposed XGboost method and [27].

Finally, we evaluate the proposed method using an accuracy analysis against seven classical machine learning algorithms, in addition to the deep neural network [27]. As shown in Figure 10, the proposed method achieves an accuracy of 95.55%, while the second-best accuracy performance is 82.02 for the NB tree, showing a significant difference between the accuracy of our proposed method and the other approaches. This evaluation confirms that the proposed method is accurate and robust, even compared against other algorithms. This shows how the unambiguous steps in our approach are reliable, effective and authoritative.

We conclude that the proposed method achieves a verifiable result using several techniques. For the precise literature and comparison, we carefully chose the NSL-KDD data set, which is considered one of the most powerful benchmark datasets. Several procedures of data statistics, cleaning and verification are performed on the dataset, which are very important in order to produce a smooth learning process with no obstacles, such as over- or under-fitting issues. This stage ensures that the proposed model has unified data and increases the value of data, which helps in decision-making. Feature normalization and selection clarifies the path for clear selection and intelligent preferences, using only

5 features. Subsequently, more detailed exploration and various comparisons are carried out, based on three machine learning algorithms, i.e., DT, RF, and XGBoost, in order to test their performance with different criteria and then select the best performing algorithm for our task. This shows that the selection is dependably proven and technically verified.

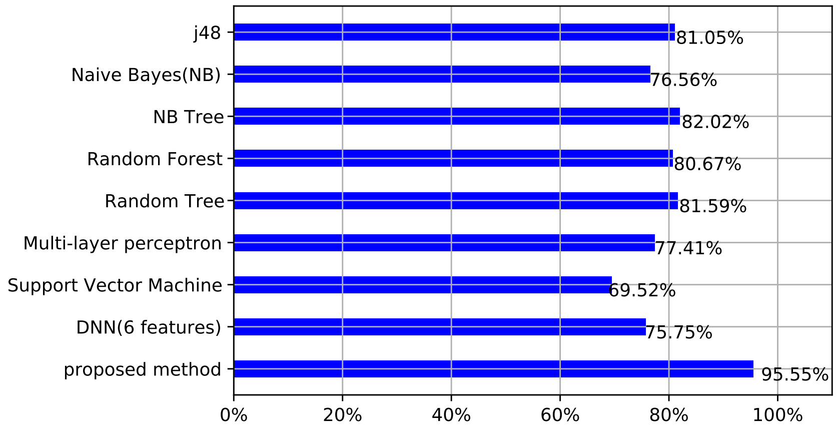

Figure 10. Accuracy performance comparison for proposed XGboost method against classical machine learning algorithms, including the DNN model in [27].

6. Conclusions and Future Work

NIDS in SDN-based machine learning algorithms has attracted significant attention in the last two decades because of the datasets and various algorithms proposed in machine learning, using only limited features for better detection of anomalies better and more efficient network security. In this study, the benchmarking dataset NSL-KDD is used for training and testing. Feature normalization, feature selection and data preprocessing techniques are used in order to improve and optimize the algorithm's performance for accurate prediction, as well as to facilitate a smooth training process with minimal time and resources. To select the appropriate algorithm, we compare three classical tree-based machine learning algorithms; Random Forest, Decision Trees and XGBoost. We examine them using a variety of evaluation metrics to find the disadvantages and advantages of using one or more. Using six different evaluation metrics, the proposed XGBoost model outperformed more than seven algorithms used in NIDS. The proposed method focused on detecting anomalies and protecting the SDN platform from attacks in real-time scenarios. The proposed methods performed two tasks simultaneously; to detect if there is an attack or not, and to determine the type of attack (Dos, probe, U2R, R2L).

In future studies, more evaluation metrics will be carried out. We plan to implement the approach using several deep neural network algorithms, such as Auto-Encoder, Generative Adversarial Networks, and Recurrent neural networks, such as GRU and LSTM. These techniques have been proven in the literature to allow convenient anomaly detection approaches in NIDS applications. Also, we plan to compare these algorithms against each other and integrate one or more neural network architectures to extract more details of how we can implement an efficient anomaly detection system in NIDS, with lower consumption of time and resources.

In addition, for a more solid basis for comparison, several benchmarking cyber security datasets, such as NSL-KDD, UNSW-NB15, CIC-IDS2017 will be conducted, in order to make sure that the selection of the proposed algorithm is not biased in any situation. These various datasets are generated in different environments and conditions, so more complex features will be available, more generalized attacks will be covered and the accuracy of the proposed algorithm will significantly increase, which could lead to a state-of-the-art approach.

Author Contributions: A.O.A. performed the experiments, analyzed the data, and wrote the paper. M.J.F.A. supervised the research and critically revised the paper. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group No (RG-1441-512).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank the Deanship of Scientific Research and RSSU at King Saud University for their technical support.

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

Abbreviations

The following abbreviations are used in this manuscript:

2-D	2 dimensional
AE	Auto-encoder
AI	Artificial Intelligence
AUC	Area Under Curve
BDA	Big Data analytics
Bi-LSTM	Bi-directional Long-Short-Term-Memory
BoW	Bag of Word
CART	Classification And Regression Tree
CNN	Convolutional Neural Networks
C	Class
DOS	denial-of-service attack
DT	Decision Tree
GAN	Generative Adverbial Network
GPU	Graphical Processing Unit
GRU	Gated Recurrent Unit
HIDS	host-based intrusion detection system
ID3	Iterative Dichotomiser 3
IDS	Intrusion Detection System
IoT	Internet of Things
LDA	Latent Dirichlet Allocation
LSTM	Long-Short-Term-Memory
Min-Max	Minimum-Maximum
NBA	Network Behavior Analysis
NB	Naive Bayes
NIDS	Network Intrusion Detection System
PCA	Principle Component Analysis
P	Precision
R2L	Remote to Local
RF	Random Forest
ROC	Receiver Operating Characteristic
R	Recall
SDN	Software Defined Network
SVM	Support Vector Machine
TSDL	Two Stages Deep Learning
U2R	User to Root
WIDS	wireless intrusion detection system
XGBoost	eXtreme Gradient Boosting

References

1. Hurley, T.; Perdomo, J.E.; Perez-Pons, A. HMM-Based Intrusion Detection System for Software Defined Networking. In Proceedings of the 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 18–20 December 2016; pp. 617–621. [CrossRef]
2. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A Deep Learning Approach to Network Intrusion Detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
3. Gómez, J.; Gil, C.; Baños, R.; Márquez, A.L.; Montoya, F.G.; Montoya, M.G. A Pareto-based multi-objective evolutionary algorithm for automatic rule generation in network intrusion detection systems. *Soft Comput.* **2013**, *17*, 255–263. [CrossRef]
4. Sangeetha, S.; Gayathri devi, B.; Ramya, R.; Dharani, M.K.; Sathya, P. Signature Based Semantic Intrusion Detection System on Cloud. In *Information Systems Design and Intelligent Applications*; Mandal, J.K., Satapathy, S.C., Kumar Sanyal, M., Sarkar, P.P., Mukhopadhyay, A., Eds.; Springer: New Delhi, India, 2015; pp. 657–666. [CrossRef]
5. Dey, S.K.; Rahman, M.M. Effects of Machine Learning Approach in Flow-Based Anomaly Detection on Software-Defined Networking. *Symmetry* **2020**, *12*, 7. [CrossRef]
6. Gao, M.; Ma, L.; Liu, H.; Zhang, Z.; Ning, Z.; Xu, J. Malicious Network Traffic Detection Based on Deep Neural Networks and Association Analysis. *Sensors* **2020**, *20*, 1452. [CrossRef] [PubMed]
7. Nobakht, M.; Sivaraman, V.; Boreli, R. A Host-Based Intrusion Detection and Mitigation Framework for Smart Home IoT Using OpenFlow. In Proceedings of the 2016 11th International Conference on Availability, Reliability and Security (ARES), Salzburg, Austria, 31 August–2 September 2016; pp. 147–156. [CrossRef]
8. Sebbar, A.; Zkik, K.; Baddi, Y.; Boulmalf, M.; Kettani, M.D.E.C.E. MitM detection and defense mechanism CBNA-RF based on machine learning for large-scale SDN context. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 5875–5894. [CrossRef]
9. Ngo, D.M.; Pham-Quoc, C.; Thinh, T.N. Heterogeneous Hardware-based Network Intrusion Detection System with Multiple Approaches for SDN. *Mob. Netw. Appl.* **2020**, *25*, 1178–1192. [CrossRef]
10. Bag, S.; Gupta, S.; Wood, L. Big data analytics in sustainable humanitarian supply chain: Barriers and their interactions. *Ann. Oper. Res.* **2020**. [CrossRef]
11. Ali, F.; Ali, A.; Imran, M.; Naqvi, R.A.; Siddiqi, M.H.; Kwak, K.S. Traffic accident detection and condition analysis based on social networking data. *Accid. Anal. Prev.* **2021**, *151*, 105973. [CrossRef]
12. Sarkar, S.K.; Roy, S.; Alsentzer, E.; McDermott, M.B.A.; Falck, F.; Bica, I.; Adams, G.; Pfohl, S.; Hyland, S.L. Machine Learning for Health (ML4H) 2020: Advancing Healthcare for All. *Proc. Mach. Learn. Res.* **2020**, *136*, 1–11.
13. Ali, F.; El-Sappagh, S.; Islam, S.R.; Kwak, D.; Ali, A.; Imran, M.; Kwak, K.S. A smart healthcare monitoring system for heart disease prediction based on ensemble deep learning and feature fusion. *Inf. Fusion* **2020**, *63*, 208–222. [CrossRef]
14. Lazarevic, A.; Kumar, V.; Srivastava, J. Intrusion Detection: A Survey. In *Managing Cyber Threats: Issues, Approaches, and Challenges*; Springer: Boston, MA, USA, 2005; pp. 19–78. [CrossRef]
15. Sultana, N.; Chilamkurti, N.; Peng, W.; Alhadad, R. Survey on SDN based network intrusion detection system using machine learning approaches. *Peer Netw. Appl.* **2019**, *12*, 493–501. [CrossRef]
16. Modi, C.; Patel, D.; Borisaniya, B.; Patel, H.; Patel, A.; Rajarajan, M. A survey of intrusion detection techniques in Cloud. *J. Netw. Comput. Appl.* **2013**, *36*, 42–57. [CrossRef]
17. Bawany, N.Z.; Shamsi, J.A.; Salah, K. DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions. *Arab. J. Sci. Eng.* **2017**, *42*, 425–441. [CrossRef]
18. Ye, J.; Cheng, X.; Zhu, J.; Feng, L.; Song, L. A DDoS Attack Detection Method Based on SVM in Software Defined Network. *Secur. Commun. Netw.* **2018**, *2018*, 9804061. [CrossRef]
19. Latah, M.; Toker, L. Artificial intelligence enabled software-defined networking: A comprehensive overview. *IET Netw.* **2019**, *8*, 79–99. [CrossRef]
20. Heorhiadi, V.; Reiter, M.K.; Sekar, V. Simplifying Software-Defined Network Optimization Using SOL. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16), Santa Clara, CA, USA, 16–18 March 2016; pp. 223–237.
21. Martini, B.; Adami, D.; Sgambelluri, A.; Gharbaoui, M.; Donatini, L.; Giordano, S.; Castoldi, P. An SDN orchestrator for resources chaining in cloud data centers. In Proceedings of the 2014 European Conference on Networks and Communications (EuCNC), Bologna, Italy, 23–26 June 2014; pp. 1–5. [CrossRef]
22. Raza, S.; Huang, G.; Chuah, C.; Seetharaman, S.; Singh, J.P. MeasuRouting: A Framework for Routing Assisted Traffic Monitoring. In Proceedings of the 2010 Proceedings IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010; pp. 1–9. [CrossRef]
23. Akyildiz, I.F.; Lee, A.; Wang, P.; Luo, M.; Chou, W. A roadmap for traffic engineering in SDN-OpenFlow networks. *Comput. Netw.* **2014**, *71*, 1–30. [CrossRef]
24. Manso, P.; Moura, J.; Serrão, C. SDN-Based Intrusion Detection System for Early Detection and Mitigation of DDoS Attacks. *Information* **2019**, *10*, 106. [CrossRef]
25. Wang, P.; Chao, K.; Lin, H.; Lin, W.; Lo, C. An Efficient Flow Control Approach for SDN-Based Network Threat Detection and Migration Using Support Vector Machine. In Proceedings of the 2016 IEEE 13th International Conference on e-Business Engineering (ICEBE), Macau, China, 4–6 November 2016; pp. 56–63. [CrossRef]
26. Revathi, S.; Malathi, A. A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *Int. J. Eng. Res. Technol.* **2013**, *2*, 1848–1853.

27. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep learning approach for Network Intrusion Detection in Software Defined Networking. In Proceedings of the 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 26–29 October 2016; pp. 258–263. [CrossRef]
28. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
29. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks. In Proceedings of the 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), Montreal, QC, Canada, 25–29 June 2018; pp. 202–206. [CrossRef]
30. Ikram, S.T.; Cherukuri, A.K. Improving Accuracy of Intrusion Detection Model Using PCA and optimized SVM. *J. Comput. Inf. Technol.* **2016**, *24*, 133–148. [CrossRef]
31. Zolotukhin, M.; Hämäläinen, T.; Kokkonen, T.; Niemelä, A.; Siltanen, J. Data Mining Approach for Detection of DDoS Attacks Utilizing SSL/TLS Protocol. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*; Balandin, S., Andreev, S., Koucheryavy, Y., Eds.; Springer: Cham, Switzerland, 2015; pp. 274–285.
32. Mehr, S.Y.; Ramamurthy, B. An SVM Based DDoS Attack Detection Method for Ryu SDN Controller. In Proceedings of the 15th International Conference on Emerging Networking EXperiments and Technologies, Orlando, FL, USA, 9–12 December 2019; pp. 72–73. [CrossRef]
33. Khan, F.A.; Gumaei, A.; Derhab, A.; Hussain, A. A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection. *IEEE Access* **2019**, *7*, 30373–30385. [CrossRef]
34. Malaiya, R.K.; Kwon, D.; Suh, S.C.; Kim, H.; Kim, I.; Kim, J. An Empirical Evaluation of Deep Learning for Network Anomaly Detection. *IEEE Access* **2019**, *7*, 140806–140817. [CrossRef]
35. Yang Jia, M.W.; Wang, Y. Network intrusion detection algorithm based on deep neural network. *IET Inf. Secur.* **2019**, *13*, 48–53. [CrossRef]
36. Yang, Y.; Zheng, K.; Wu, B.; Yang, Y.; Wang, X. Network Intrusion Detection Based on Supervised Adversarial Variational Auto-Encoder With Regularization. *IEEE Access* **2020**, *8*, 42169–42184. [CrossRef]
37. Andresini, G.; Appice, A.; Mauro, N.D.; Loglisci, C.; Malerba, D. Multi-Channel Deep Feature Learning for Intrusion Detection. *IEEE Access* **2020**, *8*, 53346–53359. [CrossRef]
38. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794. [CrossRef]