Original Research Article

COMPARATIVE ANALYSIS OF MACHINE LEARNING ALGORITHMS FOR INTRUSION DETECTION

ABSTRACT

This study undertakes a comparative examination of machine learning algorithms used for intrusion detection, addressing the escalating challenge of safeguarding networks from malicious attacks in an era marked by a proliferation of network-related applications. Given the limitations of conventional security tools in combatting intrusions effectively, the adoption of machine learning emerges as a promising avenue for bolstering detection capabilities. The research evaluates the efficacy of three distinct machine learning algorithms—Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Naive Bayes—in identifying diverse attack categories, including Denial of Service, Probe, Remote to Local, and User to Root.

Conducted on the NSL-KDD dataset, the analysis unveils CNN and RNN as superior performers compared to Naive Bayes, particularly in terms of detection accuracy. These findings extend value to both researchers and practitioners in the realm of intrusion detection systems, offering insights into optimal algorithmic choices. Furthermore, the study's implications resonate within broader contexts, such as the advancement of secure automation in industrial environments and the realm of automobile automation. Overall, this research contributes to the ongoing efforts to fortify network security and promotes the development of safer technological landscapes.

Keywords: (ABS) Intrusion Detection System, Machine Learning, Neural Networks (CNN), Recurrent Neural Networks (RNN), and Naive Bayes

I INTRODUCTION

The network of computers sharing the same information is one of the technological catalysts that changed the technological world. Presently most devices need interconnectivity with one another to function well. This interconnectivity of devices has opened doors for malicious individuals and programs to gain unauthorized access to computer networks. Network intrusion is one of the most persistent attacks on the computer network for over many decades now. The first internet-wide attack and penetration were on 2nd November 1988 in the 'Send mail program'. The safety of computers connected over the network is being affected by Network intrusion like; Backdoor attacks, distributed denial of service, denial of service, and other vector attacks.

In common parlance, network intrusion can be defined as any unauthorized access to a computer network. it is a clandestine activity that involves compromising the security of a computer network in a view of exploiting its weakness for personal gains. According to Delamore (2015), it is an unauthorized activity on a digital network. Network intrusions often involve stealing valuable network resources and almost always jeopardize the security of networks and/or their data. Also, Gragido (2016) noted that most of the unwanted activity tends to absorb network resources meant for other uses, thereby threatening the security of the network and the data involved.

In order to steam down the severity of network intrusions, the use of network intrusion detection systems has proved to be effective. an intrusion detection system (IDS) is a hardware or software that functions as a watchdog over the network in order to monitor, detect and analyze the network traffic against unauthorized intrusion and therefore raised an alarm to the network administrator when a threat is detected thereof (Quamar Niyaz, Ahmad and Mansoor, 2019). Some IDS's which are capable of responding to detected intrusion upon discovery are known as intrusion prevention systems (IPS).

A network-based intrusion detection system (NIDS) detects malicious traffic on a network. NIDS usually requires promiscuous network access in order to analyze all traffic, including all unicast traffic. NIDS are passive devices that do not interfere with the traffic they monitor. According to Asif and Yakoob (2017) Intrusion Detection mechanism attempts to discover unauthorized access to a computer network by analyzing traffic on the network for signs of malicious activity. Also, Denning (2008) observed that Intrusion Detection System offers a vital aspect of network security as it offers unlimited protection to the network, by detecting both successful and unsuccessful attempts of intrusion. According Karatas and Sahingoz (2018) Intrusion detection systems are broadly divided into two approaches, supervised and unsupervised. Initial supervised models used data mining techniques to evaluate the behavior of intrusions and normal activity.

There are several machine learning approaches used in developing an intrusion detection system. Most methods have their various advantages and pitfalls. There are equal areas where most of them perform better than others. The most used machine learning approach to intrusion detection system is convolutional neural networks and Recurrent neural networks. According to Sharafaldin (2018), A CNN model consists of convolutional layers, max-pooling layers, and a fully connected layer. CNN is the most commonly

used deep learning algorithm for image training. Also, a recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data. Javaid et al, (2016) noted that a recurrent neural network is a class of neural networks that allow previous outputs to be used as inputs while having hidden states. it is a neural network that permits a special type of artificial neural network adapted to work for time series data or data that involves sequences Kevric, (2017). Ordinary feed-forward neural networks are only meant for data points, which are independent of each other. Another one to be compared is Naïve Bayes. It is based on Bayes probability theory. It is a classification technique that is based on an assumption of independence between predictors. This implies the three machine learning algorithms would be compared with each other using NSL KDD dataset.

The NSL-KDD dataset consists of several attributes which give the features of network-based intrusion detection systems. The dataset consists of 42 attributes out of which 41 attributes are features and one attribute is a class label as normal traffic or anomaly traffic.

II LITERATURE SURVEY

A cloud-based IDS framework has been proposed by There are several scholars that have published relevant documents or related articles on network intrusion detection systems; Yasir et al., (2016), work evaluate all the machine learning algorithms provided by Weka against the standard data set for intrusion detection. they made use of KddCupp99. Different measurements evaluated are False Positive Rate, precision, ROC, and True Positive Rate.

McHugh's (2000) work was based on the evolutionary testing of the 1998 and 1998 intrusion detection systems performed by the Lincoln Laboratory. The work discussed and reviewed some important issues related to the design and execution of the evaluation. The author observed that the methodologies adopted in the evaluation were ineffective; he also noted that the appropriateness of the evaluation techniques themselves is subject to empirical verification. inferred that the evaluators made less publicity of the relatively few concerning aspects, such as the validation of test data. The author noted that the paper does not make a direct technical contribution or provide solutions to the problems raised. It is imperative to note that the author feels that the public records of such an evaluation should contain sufficient information to permit replication of the results and to understand the rationale behind most of the decisions made by the investigators.

Sabhnani Maheshkumar and Gürsel Serpen (2003) made use of a few sets of machine learning algorithms as opposed to the most commonly used KDD 1999 Cup intrusion detection dataset. The authors observed that the KDD 1999 data set encountered a lot of challenges with the resultant issues like user-to-root and remote-to-local attack categories, as reported in the recent literature. According to the authors, the need to test whether other machine learning could yield different and better results is instigated by his research. Specifically, the exploration of whether certain algorithms perform better for certain attack classes and, consequently, whether a multi-expert classifier design can deliver desired performance measures is of high interest. The results of the simulation study implemented to that effect indicated that certain classification algorithms perform better for certain attack categories. creation of a specific algorithm specialized for a given attack category. Consequently, a multi-classifier model, where a specific detection algorithm is associated with an attack category for which it is the most promising, was built. Empirical results obtained through simulation indicate that noticeable performance improvements were achieved for probing, denial of service, and user-to-root attacks.

Duque and Nizam (2015), work hinged on the application of data mining algorithms in developing an effective and reliable strategy for network intrusion detection and control. The work noted that the general problem shared by current IDS is high false positives and low detection rate. The authors strategized for unsupervised machine learning modeled from k-means to propose a model for Intrusion Detection System (IDS) with a higher efficiency rate and low false positives and false negatives. in this stage of Pre-processing, other exercise carried out was converting the attributes of the dataset into numeric data and saving in a format readable. This is very important as observed by Jiawei and Michaeline (2006) because k-means is compatible or operates on numerical data set. also it could work on alphanumeric data were converted to numeric values starting from 0.001, 0.002, and so on. Smaller values (instead of 1, 2, etc.) were used to make sure that it will not affect the computations. The two authors found that Results of k-means clustering showed that a higher efficiency rate is achieved when the correct number of clusters is applied, and increasing or decreasing the cluster beyond the number of data types only lessens the efficiency of the model. From the inference drawn from the data used by the scholars, the effective rate of the use of data mining in intrusion control was 53% which is below the incidence of cyber-attacks and network intrusion, therefore there is a need for an improvement in the work of Duque and Nizam (2015).

Kumar et al. (2016) proposed and evaluated Network Based Intrusion Detection Systems based on machine learning to detect threats to the network. In this study, different supervised machine learning classifiers are constructed using datasets including labeled examples of network traffic features created by various benign and malicious applications. The main goal of this study is Android-based malware due to the increase in mobile malware and its popularity among users. For testing the proposed approach, traffic was generated. Several malware examples such as Premium SMS sender, backdoor, spammer, bots, ransomware, information stealing and fake antivirus were used to generate this traffic. According to the obtained results, the proposed approach was able to detect unknown and known attacks up to 99.4% accuracy. This study can be improved by enlarging the created dataset and integrating it into the existing intrusion detection systems mentioned.

According to Qassim et al. (2016), anomaly-based intrusion detection system (AIDS) can identify the network traffic that is detected as malicious. It raises an alarm each time when it detects an activity that is different from the normal behaviors. Therefore, managing IDS alarms and distinguishing false positives from true alarms becomes a major challenge. This study proposed an approach consisting of two steps. Firstly, they suggested a set of network traffic features that are supposed to be the most relevant features in detecting anomalies in the network. Secondly, an AIDS alarm classifier proposed to classify activities automatically by a packet header-based anomaly detection system. According to the authors, the proposed system based on machine learning algorithms is effective and efficient in terms of classifying malicious activities. This study can be improved using various machine learning techniques to increase the accuracy rate.

Nisioti et al. (2018) conducted a survey inquiry on unsupervised and hybrid intrusion detection algorithms. The findings reveal that contemporary intrusion detection systems have progressed from simple detection to correlation and attribution. As a result, the attacker can be identified using modern data analytics tools. Furthermore, the study proposes expanding the existing attack classes by

adding three new categories pertaining to incoming network traffic. But I'm not sure what the three additional categories signify. However, the outcomes of each approach employed in IDS vary due to the dataset and methodologies used.

Mazini et al. (2019) proposes a new hybrid network-based IDS approach to detect anomaly by using AdaBoost and artificial bee colony (ABC) algorithms. Feature selection was made using the ABC algorithm. The AdaBoost algorithm was used to evaluate and classify the selected features. The proposed approach was applied to NSL-KDD and ISCXIDS2012 datasets to evaluate the accuracy of the method. 98.9% accuracy rate is achieved. According to the authors, the proposed method outperformed other IDSs on the same dataset. In the future studies, accuracy can be further improved and performance evaluation can be made on different datasets.

Meftah et al. (2019) implemented an anomaly-based network intrusion detection approach using the UNSW-NB15 dataset. Their approach consists of two main stages. They use Recursive Feature Elimination and Random Forests among other techniques to select important features for machine learning purposes. Then they perform a binary classification to detect abnormal traffic using different data mining techniques such as Support Vector Machine, Gradient Boost Machine and Logistic Regression. They achieved the highest accuracy result of 82.11% with the Support Vector Machine. They then feed the output of the SVM into a set of polynomial classifiers to increase the accuracy of detecting attack types. In particular, they evaluated the performance of Naive Bayes, Decision Trees and polynomial SVM. The application of the two-stage hybrid classification increased the accuracy of the results up to 86.04%. This work can be further developed on different datasets by developing a new classification algorithm or using deep learning techniques.

Ghaffarian et al. (2020) provides a comprehensive review of the use of recurrent neural networks (RNNs) in network intrusion detection systems (NIDSs). The authors discuss the advantages and limitations of using RNNs in NIDSs, as well as the different architectures of RNNs that have been applied in this context The strengths of the study include its thorough analysis of the literature on RNN-based NIDSs, as well as the clear and concise presentation of the advantages and disadvantages of using RNNs in this context. The paper also provides a useful discussion of the different types of RNN architectures that have been used in NIDSs, and the specific applications of these architectures. However, the review is focused only on RNN-based NIDSs, and does not cover other types of machine learning or deep learning algorithms that may also be used in this context. Additionally, while the paper provides a comprehensive review of the literature, it does not provide a detailed analysis of the specific datasets or experimental setups used in the studies reviewed. Finally, the review is limited to studies published up until 2020, and may not include more recent developments in RNN-based NIDSs.

Alamiedy et al. (2020) conducted research to enhance intrusion detection anomalies using the algorithm Grey Wolf Optimization (GWO). Based on the experiment results, it was revealed that the methods used had good accuracy for DoS (93.64%), Probe (91.01%), R2L (57.72%), and U2R (53.7%). However, the research has not tested other types of intrusion.

Vinayakumar et al. (2019) explored a form of deep learning called a "deep neural network," a type of deep learning model, to develop a flexible and effective IDS to detect and classify unforeseen and unpredictable cyberattacks. According to the authors, this is wide because the continuous change in network behavior and rapid evolution of attacks make it necessary to evaluate various datasets that have been generated over the years through static and dynamic approaches. The study dwelt on the comprehensive evaluation of experiments with DNNs and other classical machine learning classifiers shown on various publicly available benchmark malware datasets. The optimal network parameters and network topologies for DNNs are chosen through the following hyper parameter selection methods with the KDDCup 99 dataset: All the experiments with DNNs are run to 1,000 epochs, with the learning rate varying in the range [0.01-0.5]. The DNN model, which performed well on KDDCup 99, is applied to other datasets, such as NSL-KDD, UNSW-NB15, Kyoto, WSN-DS, and CICIDS 2017, to conduct the benchmark. The finding and the recommendation from the study were that DNN model learns the abstract and high-dimensional feature representation of the IDS data by passing them into many hidden layers. The authors proposed for highly scalable and hybrid DNNs framework called scale-hybrid-IDS-AlertNet which can be used in real-time to effectively monitor the network traffic and host-level events to proactively alert possible cyber-attacks.

Mambwe and Yanxia (2020) proposed a feed-forward deep neural network (FFDNN) wireless IDS system using a wrapper-based feature extraction unit (WFEU). The method of data collection adopted by the work was the extra-tree algorithm in order to generate a reduced optimal feature vector. The effectiveness and efficiency of the WFEU-FFDNN were studied based on the UNSW-NB15 and the AWID intrusion detection datasets. Furthermore, the WFEU-FFDNN is compared to standard machine learning (ML) algorithms that include Random Forest (RF), Support Vector Machine (SVM), Nave Bayes (NB), Decision Tree (DT), and k-Nearest Neighbor (kNN). The experimental studies include binary and multiclass types of attacks. In their findings, they discovered that WFEU-FFDNN has greater detection accuracy than other approaches that have been used by several scholars. The author noted that the UNSW-NB15 and the WFEU generated a maximum vector consisting of 22 attributes. The percentage as noted by Mambwe and Yanxia (2020) was of accuracies 87.10% and 77.16% for the binary and multiclass classification schemes, respectively.

Park et al. (2021) carried out an experiment on the Leipzig Intrusion Detection Dataset (LID-DS), which is a host-based IDS dataset created in 2018. In addition, an intrusion detection model consisting of host-based preprocessing, vector-to-image processing, training, and testing steps is proposed to improve the performance of the system. In the training and testing steps, a Siamese convolutional neural network (Siamese-CNN) was constructed using a learning technique consisting of several steps with high performance using a small amount of data. Siamese-CNN determines the type of attack based on the similarity score of each attack sample converted to an image. Accuracy is calculated using a few shot-learning techniques. For performance evaluation, the performances of the Vanilla Convolutional Neural Network (Vanilla-CNN) and the Siamese-CNN were compared. According to the accuracy, precision, recall, and F1-score indicators, the proposed Siamese-CNN model showed an increase of approximately 6% compared to the vanilla-CNN model. The proposed model can be developed by working on increasing the accuracy of intrusion detection for known or unknown cyber-attacks by optimizing the hyper parameter values.

The study by Li, Xu, Wang, and Hu (2021) proposes a new approach for intrusion detection in software-defined networks (SDNs) using a combination of decision tree and Naive Bayes algorithm. The authors aim to enhance the accuracy and efficiency of intrusion detection in SDNs while reducing false alarms. To achieve this, the authors developed a three-stage intrusion detection system (IDS) based on a combination of decision tree and Naive Bayes algorithm. The first stage involves data preprocessing, where the raw

network traffic data is filtered and transformed into a feature vector. The second stage is feature selection, where the most relevant features for intrusion detection are selected using a feature selection algorithm. In the final stage, the decision tree and Naive Bayes algorithm are used to classify network traffic data as either normal or anomalous. The authors evaluated the proposed approach using a dataset that contains six types of network attacks, namely, distributed denial-of-service (DDoS), port scan, SQL injection, cross-site scripting (XSS), brute-force attack, and remote-to-local (R2L) attack. The authors compared the proposed approach's performance to the Naive Bayes algorithm alone, and the results show that the proposed approach achieves a higher accuracy rate than the Naive Bayes algorithm alone. The authors also conducted a comparison of the proposed approach with other intrusion detection methods, and the results show that the proposed approach outperforms the other methods in terms of accuracy rate. However, the study only compared the proposed approach to the Naive Bayes algorithm without comparing it to other existing intrusion detection methods. It is essential to compare the proposed approach to other established methods to assess its effectiveness relative to other intrusion detection methods.

Similarly, the study by Sadiq, Abbas, and Amin (2021) proposes an intelligent approach for network intrusion detection based on the decision tree and Naive Bayes algorithm. The authors aimed to design an approach to enhance the accuracy of intrusion detection while reducing the false alarm rate. To achieve this, the authors developed a three-stage intrusion detection system (IDS). In the first stage, the dataset is preprocessed, where the raw network traffic data is filtered and transformed into a feature vector. In the second stage, feature selection is performed to identify the most relevant features for intrusion detection using a correlation-based feature selection (CFS) algorithm. In the final stage, the decision tree and Naive Bayes algorithm are used to classify the network traffic data as either normal or anomalous. The authors evaluated the performance of the proposed approach using the benchmark KDD99 dataset, which contains four types of network attacks, namely, denial-of-service (DoS), probe, user to root (U2R), and remote to local (R2L) attacks. The authors compared the performance of the proposed approach with other existing intrusion detection methods, such as Naive Bayes, J48, and Random Forest. The results showed that the proposed approach achieved a higher accuracy rate than the other intrusion detection methods. Moreover, the proposed approach had a low false alarm rate, indicating its effectiveness in detecting network intrusions. The authors concluded that the proposed approach can be used as an effective technique for network intrusion detection. However, the study did not investigate the impact of varying the number of features used in the intrusion detection process. It would be interesting to explore how the number of features affects the accuracy and false alarm rate of the proposed approach.

The work by Chaudhari and Chaudhari (2021) presents a study on the use of the Naive Bayes algorithm for network intrusion detection. The authors start by discussing the importance of intrusion detection in network security and how machine learning algorithms can be used to improve its efficiency. They then provide a brief overview of the Naive Bayes algorithm and its application in classification problems. The authors used the KDD Cup 99 dataset to evaluate the performance of the Naive Bayes algorithm for network intrusion detection. They preprocessed the data by removing duplicates, normalizing the features, and splitting it into training and testing sets. They then trained the Naive Bayes classifier on the training set and evaluated its performance on the testing set using metrics such as accuracy, precision, recall, and F1-score.

The results of the study showed that the Naive Bayes algorithm achieved an accuracy of 93.76% for network intrusion detection. The precision, recall, and F1-score were also reported, with precision ranging from 0.84 to 1.00, recall ranging from 0.72 to 1.00, and F1-score ranging from 0.77 to 1.00. The authors compared the performance of the Naive Bayes algorithm with other machine learning algorithms such as Support Vector Machines (SVMs) and Artificial Neural Networks (ANNs), and found that the Naive Bayes algorithm outperformed these algorithms in terms of accuracy. The authors concluded that the Naive Bayes algorithm is a promising technique for network intrusion detection and can achieve high accuracy with low computational complexity. However, they acknowledged that the study has limitations, such as the use of a single dataset and the need for further evaluation on larger and more diverse datasets. They suggested that future work could investigate the use of ensemble methods and feature selection techniques to further improve the performance of the Naive Bayes algorithm for network intrusion detection. However, the study did not consider the impact of feature selection on the performance of the Naive Bayes algorithm. Feature selection is an important step in the preprocessing of data for machine learning algorithms, and it can significantly affect the accuracy and efficiency of the algorithm. Future studies could investigate the use of feature selection techniques to improve the performance of the Naive Bayes algorithm for network intrusion detection.

III METHODOLOGY

A research design includes the structure of a study and the strategies for conducting that study (Kerlinger, 1973). AlsoFlesis (1999) noted that experimental designs have been developed to reduce biases of all kinds as much as possible. This research adopts an experimental research design. This method or design was adopted in order to determine the strength of the three machine learning (CNN, Naïve Bayes and RNN) approach which this study seek to compare. Additionally, this research work is interested in the comparative analysis of difference machine learning approach to network intrusion detection system. Determining the most detection system for anomaly in network plays a very important role in making recommendation to network security issues. There several steps involve in building a ML model such as data preparation, data preprocessing, feature selection, data visualization, clustering, model training and model validation. The following algorithms are selected for the study purposes are Logistic Regression, Naive Bayes, CNN, and Random Forest.

3.1 Source of Dataset

The NSL-KDD dataset consists of several attributes which give the features of network-based intrusion detection systems. The dataset consists of 42 attributes out of which 41 attributes are features and one attribute is a class label as normal traffic or anomaly traffic as shown in Table 1. This makes the detection more accurate and realistic as the machine learning model is also verified for unknown attacks.

Table 1:Attributes of NSL-KDD dataset

SN	NAME OF FEATURES	SN	NAME OF FEATURES
1	Duration	23	Count
2	protocol_type	24	Srv_count
3	SERVICE	25	serror_rate
4	FLAG	26	srv_serror_rate
5	src_bytes	27	rerror_rate
6	DST BYTES	28	srv_rerror_rate
7	LAND	29	same_srv_rate
8	wrong_fragt	30	diff_srv_rate
9	URGENT	31	srv_diff_h_rate
10	HOT	32	host_count
11	num_fail_login	33	host_srv_count
12	Log in	34	h_same_sr_rate
13	nu_comprom	35	h_diff_srv_rate
14	root_shell	36	h_src_port_rate
15	su_attempted	37	h_srv_d_h_rate
16	Num root	38	h_serror_rate
17	nu_file_creat	39	h_sr_serror_rate
18	Num shells	40	h_rerror_rate
19	nu_access_files	41	h_sr_rerror_rate

The NSL KDD dataset is further divided into train and test data as shown in Table 2. The train data set is used to train the machine learning ML model and the test dataset is given as inputs to the trained model for validation purposes. The advantage of this dataset is that it does not include any redundant records in the training data, so the classifiers are accurate and not biased toward more frequent records.

Table 2**:** Distribution of instance in NSL-KDD dataset for training and testing

Types of Attack	**Number of Records**	
	Training dataset	**Test dataset**
Normal	67343	9711
Denial of service	45927	7456
Probe	11656	2421
Remote to Local	995	2756
User to root	52	200
Total	125973	22544

3.2 Model Formulation Procedures

Figure 1 present the model formulation procedures. These procedures include dataset collection, data preprocessing, transformation of dataset, derived model and prediction/result evaluation.

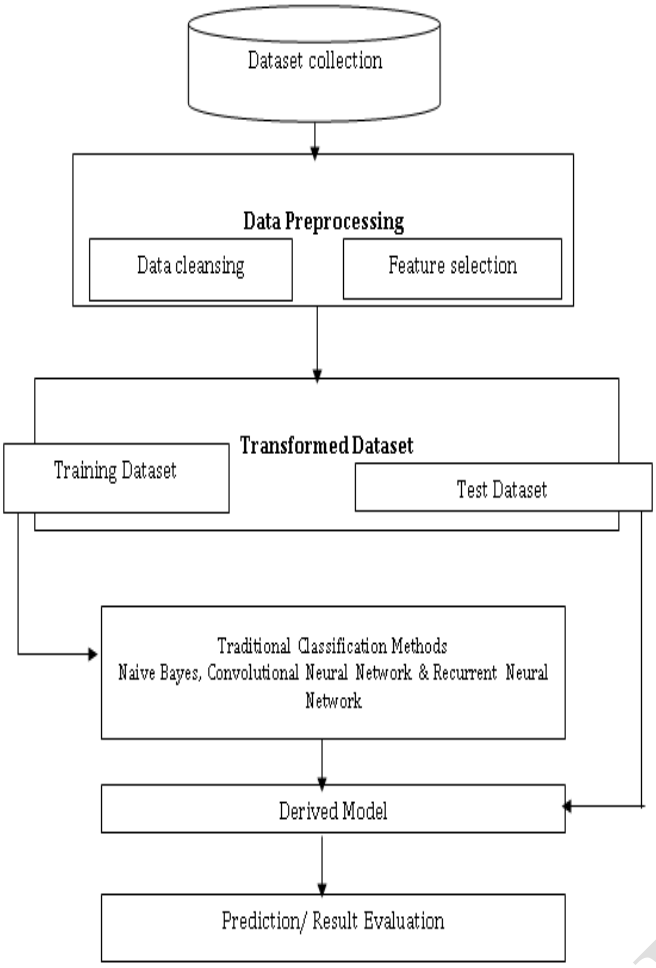

Figure 1: Model Formulation Procedures

Dataset Collection

The initial stage in every data mining assignment is to obtain raw data from a database that will be valuable for study. The type of dataset required for any research endeavor is determined by the nature of the investigation. The dataset can be obtained from either primary or secondary sources, and appropriate data gathering procedures for the research can be employed at this point.

Data Preprocessing

The technique of data pre-processing helps to eliminate irrelevant data and missing values. Here, the One Hot Encoding technique is used to convert nominal values into binary values in the dataset. it is important to note that Data preprocessing is an important step in Machine Learning as the quality of data and information have an effect on the outcome. Some data preprocessing techniques are; Handling Null values, Standardization, Handling Categorical Variables, One-Hot Encoding, and Multicolor linearity. The real-world data is incomplete, inconsistent, lacking in certain behavior, and likely to contain many errors. Data pre-processing is proven to resolve these errors like false positives, false negatives, etc.

Feature Selection

Feature selection is one of the methods which has a model outcome. The features that are used to train machine learning models have a huge influence on the performance because partially relevant or irrelevant features can negatively impact the performance. The process of selecting the features manually or automatically from the dataset mainly contributes to the prediction or output of models which are interested. The technique used here for feature selection is attribute ratio. After feature selection data standardization is done which is the process that converts the structure of disparate datasets into a Common Data Format. Standardization is required as many distance-based algorithms are used.

Transformed Dataset

At this point, a more refined dataset is gotten after the preprocessing was done which was split into training and testing dataset using 10-fold cross validation technique. The training dataset is for training or building the required model using Naïve Bayes, Convolutional Neural Network and Recurrent Neural Network, the test dataset is for testing the derived model.

- Classification Methods: Naive Bayes, convolutional neural network and Recurrent Neural Network algorithms; these are the algorithms use for the development of propose predictive model using the training dataset.
- Derived Model: At this point, new predictive model that was trained using the Naive Bayes, Convolutional Neural Network and Recurrent Neural Network algorithms will be formed. Therefore, the test data set will be feed into this model to see how the predictive result of the two algorithms look like.

- Prediction/ Result Evaluation: When the derived model is tested it will produce a result which will be observed, studied and evaluated.

3.3 Algorithm for the Model

1. Load required packages
2. Assign features
3. Read dataset
4. Check and confirm statistic reports
5. Classify labels into attack class
6. create a dataframe with multi-class labels
7. Normalize data
8. Encode multiclass label (for CNN and RNN)
9. Split dataset into test (20%) and train data (80%)
10. Reshape data (features and labels)
11. Training for CNN/RNN
11.1 Initialize models instance
11.2 Define layers
11.3 define loss function, optimizer, metrics and then compile
11.4 Fit the model(s)
12. Training for Naive Bayes
11.1 Initialize model using Gaussian Naive Bayes classifier
11.2 Fit the model
13. Evaluate the trained models
14. Output the training and prediction metrics
15. Plot graph

3.4 Performance Evaluation Metrics

The study adopts three performance evaluation metrics like; The first is "Accuracy". The accuracy denotes the number of correctly predicted data points out of all the data points and it is represented in terms of percentage. The accuracy is calculated in this research using the following standard formula;

Accuracy = $= \frac{TP + TN}{TP + TN + FP + FN}$

it, therefore, stands that False positive (FP): It defined as the number of detected attacks that are actually normal, and False negative (FN): means the wrong prediction i.e. it detects the instances as normal but in actual its an attack. True positive (TP): is an instance that is correctly predicted as normal, and True negative (TN): is an instance that is correctly classified or detected as an attack

Then the second evaluation metric is "Precision". The precision denotes the measure which estimates the probability that a positive prediction is correct, and the formula is as shown in the first equation in Accuracy rate.

Precision = $\frac{TP + TN}{TP + FP}$

Also, the third measure of the performance metric is recall. The recall is the proportion of instances belonging to the positive class that is correctly predicted as positive,

Recall= $\frac{TP}{TP + FN}$

Furthermore, the **F1 Score.** The F1 score or F-measure is a measure of the test's accuracy. It considers both the precision P and the recall R of the test to compute the score.

The final one is the **Receiver Operating Characteristics (ROC)** It is used to design the curve between the true positive rate and false positive rate, and the Area Under Curve (AUC) gives the value of ROC. The more the area under the curve and more will be the value of ROC.

4 RESULT AND DISCUSSION

The section shows the performance or accuracy rate of machine learning detection techniques (CNN, RNN, and Naive Bayes) against some attacks like denial-of-service DoS, probe attacks, remote to local attacks, and user root attacks.

4.2 Result of Classification Based on DoS Attacks

A denial-of-service (DoS) attack is a malicious attack aimed at rendering a machine or network inaccessible to its intended users. The objective of a DoS attack is to disrupt the normal functioning of a system by overwhelming it with a flood of traffic or by exploiting vulnerabilities to trigger a crash. As a result, legitimate users, such as employees, members, or account holders, are unable to access the service or resource they expect, leading to disruption, inconvenience, and potential financial losses. To combat DoS attacks, machine learning algorithms can be employed to detect and mitigate such threats. In Table 3, the performance of three machine learning algorithms (CNN, RNN, and Naive Bayes) is evaluated using various metrics: accuracy, precision, recall, F1 score, and the ROC (Receiver Operating Characteristic).

Table 3: Result of Classification based on DoS attacks.

Machine Learning	Accuracy %	Precision	Recall	F1 Score	ROC
CNN	96.55	0.971	0.964	0.967	0.965
RNN	98.44	0.987	0.985	0.986	0.658
Naive Bayes	91.30	0.913	0.899	0.906	0.899

Based on these results, the CNN algorithm demonstrates the highest accuracy among the three algorithms, achieving an accuracy rate of 96.55%. Accuracy represents the percentage of correctly classified instances, indicating how well the algorithm performs overall in detecting DoS attacks. The precision score of CNN is 0.971, which indicates the proportion of correctly identified positive instances (i.e., DoS attacks) out of the total instances identified as positive. Similarly, the recall score of 0.964 represents the proportion of actual positive instances that were correctly identified by the algorithm. The F1 score, a harmonic mean of precision and recall, is 0.967 for the CNN algorithm. This score provides a balanced assessment of the algorithm's performance in terms of both precision and recall. Lastly, the ROC score for CNN is 0.965. The ROC curve measures the algorithm's ability to distinguish between positive and negative instances, with a higher ROC score indicating better discrimination capability. Comparatively, the RNN algorithm achieves a higher accuracy of 98.44%, indicating strong performance in detecting DoS attacks. However, its ROC score of 0.658 suggests that it might have lower discrimination ability than CNN.

The Naive Bayes algorithm, although having a lower accuracy of 91.30%, still demonstrates acceptable performance in detecting DoS attacks. However, its precision, recall, F1 score, and ROC score are relatively lower than both CNN and RNN. In conclusion, based on the given results, the CNN algorithm stands out for its high accuracy in detecting DoS attacks. While RNN also performs well, its lower ROC score raises some concerns. Naive Bayes, although less accurate compared to the other algorithms, still provides a reasonable performance. Ultimately, the selection of the most suitable algorithm depends on various factors, including the specific requirements of the system and the trade-offs between accuracy and other performance metrics.

4.3 Result of Classification Based on Probe Attacks

A probe attack is an invasive method for bypassing security measures by observing the physical silicon implementation of a chip. In an invasive attack, one directly accesses the internal wires and connections of a targeted device and extracts sensitive information. Table 4 depicts the result of the machine learning comparison for the probe attack as implemented. The result of each machine learning algorithm (CNN, RNN, and Naive Bayes) in detecting probe attacks is presented under the following headings: accuracy, precision, recall, F1 score, and the ROC. From the results, RNN leads in accuracy with 99.94% in detecting probe attacks.

Table 4: Result of Classification Based on Probe Attacks

Machine Learning	Accuracy %	Precision	Recall	F1 Score	ROC
RNN	99.94	0.999	1.0	1.0	1.0
CNN	99.94	0.994	0.999	0.999	0.999
Naive Bayes	90.31	0.986	0.966	0.956	0.974

The graphical representation in Figure 2 illustrates the classification results specifically for user to root attacks. Each evaluation metric is represented by a color-coded bar, with accuracy shown in blue, precision in red, recall in neo-green, F1 score in violet, and ROC in turquoise.

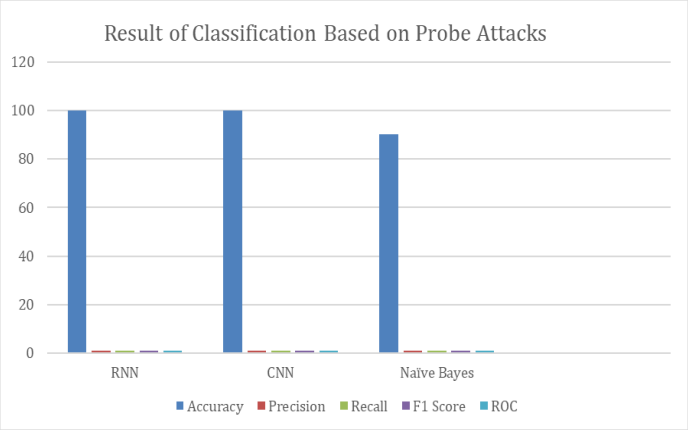

Figure 2: Result of Classification Based on Probe Attacks

4.4 Result of Classification Based on Remote to Local attack

A remote-to-local attack (r2l) has been widely known to be launched by an attacker to gain unauthorized access to a victim machine in the entire network. Similarly, a user-to-root attack (U2R) is usually launched to illegally obtain the root's privileges when legally accessing a local machine. The result of the machine-learning comparison on the remote attack is depicted in Table 5. The results of each machine learning algorithm (CNN, RNN, and Naive Bayes) in detecting remote and local attacks are presented under the following headings: accuracy, precision, recall, F1 score, and the ROC. From the results, RNN leads in accuracy, with 99.99%, in detecting both remote and local attacks.

Table 5: Result of Classification based on Remote to Local attack

Machine Learning	Accuracy %	Precision	Recall	F1 Score	ROC
RNN	99.99	0.999	1.0	0.999	1.0
CNN	99.89	0.998	0.999	0.998	0.994
Naive Bayes	98.92	0.999	0.882	0.931	0.953

Figure 3 visually displays the classification results specifically for user to root attacks. The different evaluation metrics, accuracy (blue), precision (red), recall (neo-green), F1 score (violet), and ROC (turquoise), are represented by color-coded bars in the graph.

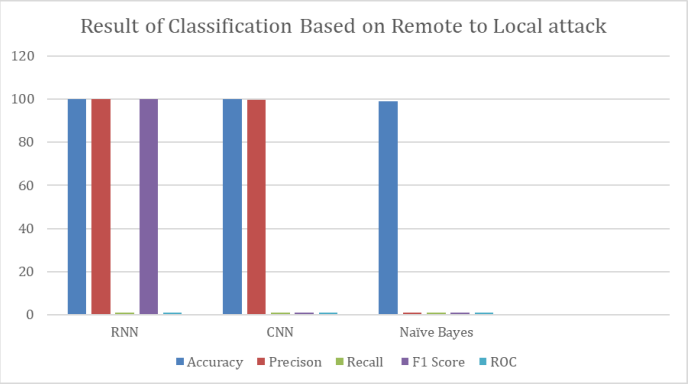

Figure 3: Result of Classification Based on Remote to Local attack

4.5 Result of Classification Based On User to Root Attack

User-to-Root (U2R) attacks have the objective of a non-privileged user acquiring root or admin-user access on a specific computer or a system on which the intruder had user-level access. Remote-to-local (R2L) attacks involve sending packets to the victim's machine. the result of machine learning comparison on User to Root attack implemented is depicted Table 6. The result of each machine learning algorithm (CNN, RNN, and Naive Bayes) in detecting user-to-root attacks is presented under the following headings: accuracy, precision, recall, F1 score, and the ROC. From the results, RNN leads in accuracy, with 99.85%, in detecting remote and local attacks.

Table 6: Result of Classification Based On User to Root Attack

ML	Accuracy %	Precision	Recall	F1 Score	ROC
RNN	99.85	0.999	0.998	0.998	0.997
CNN	99.67	1	0.998	0.998	0.936
Naive Bayes	88.85	0.999	0.999	0.960	0.948

Figure 4 graphically represents the classification results for user to root attacks. The different evaluation metrics are represented by color-coded bars: accuracy is represented by blue, precision by red, recall by neo-green, F1 score by violet, and ROC by turquoise.

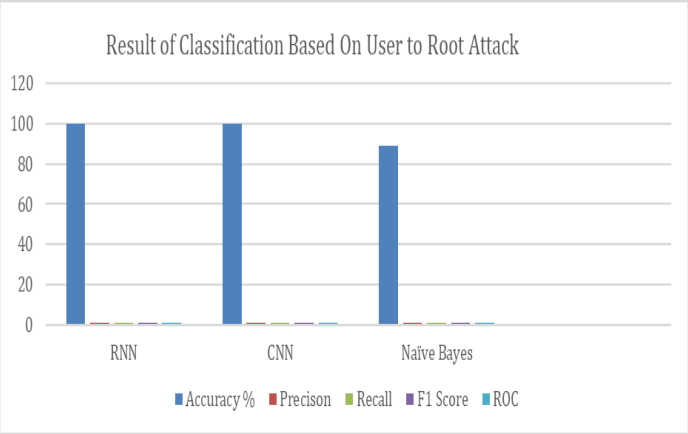

Figure 4: Result of Classification Based On User to Root Attack

4.6 Discussion of Findings

Based on the comprehensive comparative analysis conducted on the three machine learning algorithms (CNN, RNN, and Naive Bayes) in various attack scenarios, it is evident that RNN emerges as the top-performing algorithm. Its consistently high accuracy, precision, recall, F1 score, and ROC make it the preferred choice for attack detection tasks.

RNN consistently demonstrates exceptional performance across different attack scenarios, achieving high accuracy rates ranging from 96.55% to 99.99%. This indicates its ability to accurately classify attack instances and make a high percentage of correct predictions. The high accuracy is crucial in ensuring that attacks are effectively detected and mitigated, minimizing the risk of unauthorized access or damage to systems.

Precision, which measures the algorithm's ability to correctly classify positive instances while minimizing false positives, is consistently above 0.99 for RNN. This indicates a minimal false positive rate, reducing unnecessary disruptions to legitimate users or resources. By accurately identifying positive instances, RNN ensures that potential attacks are thoroughly examined before triggering any security measures. RNN also demonstrates high recall scores, indicating its effectiveness in identifying the majority of true positive instances. This is crucial in ensuring that attacks are not overlooked or missed, reducing the risk of undetected malicious activities. The balanced performance between precision and recall is reflected in the consistently high F1 scores achieved by RNN. Furthermore, RNN exhibits excellent discrimination capability as indicated by its consistently high ROC scores. This implies its ability to effectively distinguish between positive and negative instances, enhancing its reliability in attack detection. The ROC scores validate the robustness of RNN in accurately identifying attacks and minimizing false positives. While CNN also demonstrates strong performance in attack detection, with high accuracy, precision, recall, and F1 scores, its ROC scores are slightly lower compared to RNN. RNN's strength lies in its ability to capture temporal dependencies and long-term patterns, making it suitable for analyzing sequential and time-series data. However, the slightly lower ROC scores suggest a slightly weaker discrimination capability compared to CNN. On the other hand, Naive Bayes, although simpler and computationally efficient, exhibits lower accuracy and F1 scores compared to CNN and RNN. The Naive Bayes algorithm's reliance on the assumption of independence between features may limit its ability to capture complex relationships within the data. Consequently, it may struggle to accurately detect attacks in certain scenarios, leading to lower overall performance compared to the other algorithms. In conclusion, based on the comparative analysis, RNN emerges as the top-performing algorithm for attack detection tasks. Its consistent high accuracy, precision, recall, F1 score, and ROC scores indicate its superiority in identifying and classifying attacks accurately. RNN also demonstrates strong performance but may have slightly lower ROC scores. Naive Bayes, while simpler and computationally efficient, exhibits lower accuracy and F1 scores. Therefore, when selecting an algorithm for attack detection, CNN and RNN are the preferred choices, with RNN offering superior performance in most cases. However, the selection of the most suitable algorithm ultimately depends on specific requirements, available resources, and the trade-offs between different performance metrics.

V CONCLUSION

this study conducted an extensive comparative analysis of three prominent machine learning algorithms—Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Naive Bayes—in various attack scenarios. The evaluation was based on key performance metrics such as accuracy, precision, recall, F1 score, and Receiver Operating Characteristic (ROC) scores.

The results unequivocally establish RNN as the preferred algorithm for attack detection tasks. Its consistent accuracy rates, ranging from 96.55% to 99.99%, underline its exceptional ability to precisely classify and predict attacks, bolstering system security against unauthorized access and potential damage. RNN's adeptness in maintaining a fine balance between minimizing false positives and identifying true positives enhances its reliability in real-world scenarios.

Additionally, RNN's high F1 scores validate its effectiveness in simultaneously achieving both precision and recall objectives. The algorithm's robust ROC scores further affirm its capability to accurately differentiate between positive and negative instances.

While CNN also demonstrates strong performance in attack detection, its slightly lower ROC scores compared to RNN indicate a nuanced difference in discriminatory power. In contrast, Naive Bayes, while computationally efficient, exhibits inferior accuracy and F1 scores due to its assumption of feature independence.

Ultimately, this comparative analysis underscores RNN's superior performance for attack detection, with its consistent excellence in multiple performance metrics. However, the choice of algorithm should consider specific needs, available resources, and the trade-offs between different performance measures, contributing to more robust security solutions in critical systems.

VI RECOMMENDATION

This paper makes the following recommendations:

1. RNN is the more accurate machine learning algorithm for intrusion detection based on the results of the comparison. RCNN performed better when compared with four different attacks and evaluated using various evaluation metrics.
2. Both signature-based and anomaly-based intrusion detection methods are vital for detecting intrusions.
3. The combination of both methods offers better results for intrusion detection systems.

VII FUTURE WORK

The future work should focus on developing algorithms that not only detect attacks but also provide interpretable explanations for their decisions, enhancing transparency and aiding cybersecurity experts in understanding and mitigating emerging threats.

REFERENCE

Alamiedy, T. A., Anbar, M., Alqattan, Z. N., & Alzubi, Q. M. (2020). Anomaly-based intrusion detection system using multi-objective grey wolf optimisation algorithm. Journal of Ambient Intelligence and Humanized Computing, 11, 3735-3756.

Alrawais, A., Alhothaily, A., & Alhazmi, N. (2019). Deep learning for intrusion detection: A review. Journal of Ambient Intelligence and Humanized Computing, 10(10), 3981-3995. https://doi.org/10.1007/s12652-019-01412-3

Azab, M., Moustafa, A., & Ghanem, S. (2019). Intrusion detection in IoT-based networks using deep learning techniques. Journal of Network and Computer Applications, 135, 1-18. https://doi.org/10.1016/j.jnca.2019.04.007

Chaudhari, P., & Chaudhari, S. (2021). Network intrusion detection using Naive Bayes classification. International Journal of Electrical and Computer Engineering, 11(3), 2461-2468. doi: 10.11591/ijece. v11i3.pp2461-2468

Ghaffarian, S., Anpalagan, A., & Vassaki, S. (2020). Recurrent neural networks for intrusion detection systems: A review. IEEE Access, 8, 26149-26167. https://doi.org/10.1109/ACCESS.2020.2974965

Gilat, Y. Tobin, and G. Shahar, "Offering support to suicidal individuals in an online support group," Archives of Suicide Research, vol. 15, no. 3, pp. 195–206, 2011.

Ghosh, S. K., & Das, S. (2017). A survey on intrusion detection using machine learning techniques. International Journal of Computer Applications, 168(3), 20-29. https://doi.org/10.5120/ijca2017912957

Jia, Y., Liu, C., Sun, L., Zhang, M., & Han, J. (2019). Network intrusion detection based on bidirectional long short-term memory neural networks. IEEE Access, 7, 138666-138675. https://doi.org/10.1109/ACCESS.2019.2943926

Kumar, S., Viinikainen, A., & Hamalainen, T. (2016, December). Machine learning classification model for network based intrusion detection system. In 2016 11th international conference for internet technology and secured transactions (ICITST) (pp. 242-249). IEEE.

Kurin, and Shimon (2019) "Fast efficient hyperparameter tuning for policy gradients." arXiv preprint arXiv:1902.06583.

Li, Y., Xu, J., Wang, Q., & Hu, X. (2021). Intrusion detection based on decision tree and Naive Bayes algorithm in software-defined network. Journal of Ambient Intelligence and Humanized Computing, 12(6), 5781-5792. doi: 10.1007/s12652-021-03183-2

Lipton, Zachary C., John Berkowitz, and Charles Elkan. (2015) "A critical review of recurrent neural networks for sequence learning." arXiv preprint arXiv:1506.00019.

Lirim, and Dagli. (2019) "Cybersecurity as a Centralized Directed System of Systems using SoS Explorer as a Tool."2019 14th Annual Conference System of Systems Engineering (SoSE), 140-145. IEEE

Liu, S., Liu, J., Cheng, L., & Chen, Y. (2018). A deep learning based network intrusion detection system. In 2018 IEEE Conference on Communications and Network Security (CNS) (pp. 263-271). IEEE. https://doi.org/10.1109/CNS.2018.8433083

Mazini, M., Shirazi, B., & Mahdavi, I. (2019). Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. Journal of King Saud University-Computer and Information Sciences, 31(4), 541-553.

Meftah, S., Rachidi, T., & Assem, N. (2019). Network based intrusion detection using the UNSW-NB15 dataset. International Journal of Computing and Digital Systems, 8(5), 478-487.

McHuge (2000), Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincolnlaboratory."ACM Transactions on Information and System Security (TISSEC), 3 (4) (2000), pp. 262-294.

McHugh, John. (2000) "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory." ACM Transactions on Information and System Security (TISSEC) 3, no. 4: 262-294.

Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 20

Moustafa, Nour, and Jill Slay. (2015) "The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems." In 2015 4th international workshop on building analysis datasets and gathering experience returns for security (BADGERS), 25-

31. IEEE

Mukkama, and Janoski (2002). Intrusion Detection Using Ensemble of Soft Computing Paradigms. file:///C:/Users/Hp/AppData/Local/Temp/download.pdf.

Nisioti, A., Mylonas, A., Yoo, P. D., & Katos, V. (2018). From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. IEEE Communications Surveys & Tutorials, 20(4), 3369-3388.

Ouafi, K. A., Ghanemi, S., & Habbal, A. (2019). Intrusion detection using recurrent neural network with Long Short-Term Memory. Procedia Computer Science, 151, 520-527. https://doi.org/10.1016/j.procs.2019.04.068

Park, J., Shin, D., & Kang, S. (2018). Long short-term memory network-based intrusion detection system in internet of things environments. Information Sciences, 465, 90-101. https://doi.org/10.1016/j.ins.2018.06.037

Park, D., Kim, S., Kwon, H., Shin, D., & Shin, D. (2021). Host-based intrusion detection model using Siamese network. IEEE Access, 9, 76614-76623.

Quamar, Ahmad and Mansoor, (2019). A Deep Learning Approach for Network Intrusion Detection System. BICT'15: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)

Raina, R., Madhavan, A., & Ng, A. (2009). Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th annual International conferenceonmachinelearning (pp.873–880).ACM

Sadiq, S., Abbas, S., & Amin, F. (2021). An intelligent approach for network intrusion detection based on decision tree and Naive Bayes algorithm. Journal of Ambient Intelligence and Humanized Computing, 12(10), 10793-10803. doi: 10.1007/s12652-021-03667-2

Salama, Ramadan, Darwish, (2011) A hybrid machine learning approach to network anomaly detection. Information Sciences 177, 3799–3821.

Shone and Nathan (2018) A Framework for Intrusion Detection Based on Workflow Mining, American Journal of Computer Science and Technology. Vol. 2, No. 2, 2019, pp. 27-34. doi: 10.11648/j.ajcst.20190202.12

Qassim, Q., Zin, A. M., & Ab Aziz, M. J. (2016). Anomalies Classification Approach for Network-based Intrusion Detection System. Int. J. Netw. Secur., 18(6), 1159-1172.

Wen, X., Du, Y., & Wang, H. (2018). Network intrusion detection based on deep learning algorithm. 2018 IEEE International Conference on Applied System Innovation (ICASI). https://doi.org/10.1109/icasi.2018.8396955