Machine Learning Approaches Intrusion for Improving Network-Based Detection System

Seethal Prince E[1], Mahesh K M[2]

Nehru Arts and Science College, India[1],
SCMS Engineering and Technology, India[2]

Corresponding author: Seethal Prince E, Email: seethalprincee@gmail.com

The field of network security has gained paramount importance in response to the ever-growing advancements in internet and communication technologies. With the aim of safeguarding the integrity of networks and their components in the digital realm, a suite of tools including firewalls, antivirus software, and intrusion detection systems (IDS) has been deployed. Among these, network-based intrusion detection systems (NIDS) hold a pivotal role by continuously monitoring network traffic for any signs of malicious or suspicious activities. However, the relentless pace of technological progress in the past decade has led to the expansion of larger, more complex networks supporting a multitude of applications, thereby creating significant challenges in maintaining data and network node security. The existing IDSs have revealed their limitations in detecting various forms of attacks, including zero-day attacks, and mitigating false alarm rates (FAR). Consequently, the demand for cost-effective, precise, and efficient NIDS solutions is on the rise to fortify network security.

Keywords: Network Security, IDS, NIDS, Network Node Security, Zero-Day Attacks, False Alarm Rates

2023. In Satyasai Jagannath Nanda & Rajendra Prasad Yadav (eds.), *Data Science and Intelligent Computing Techniques*, 765–780. Computing & Intelligent Systems, SCRS, India. https://doi.org/10.56155/978-81-955020-2-8-66

1. Introduction

In recent years, network security has evolved into a critical area of research, driven by the ever-growing interest and advancements in Internet and communication technologies. The preservation of network security, along with its intricate technical components in the vast cyberspace, relies on a toolbox comprising firewalls, antivirus software, and intrusion detection systems (IDS)[1]. Among these, the Network-based Intrusion Detection System (NIDS) [2] stands out as a vigilant security tool, continuously monitoring network traffic for indications of malicious and suspicious activities, ensuring a robust level of protection.

The concept of IDS was initially introduced by Jim Anderson in 1980, marking the beginning of a journey that has seen the development and enhancement of a diverse array of IDS products. These advancements were geared towards meeting the evolving demands of network security. However, the relentless pace of technological progress in the past decade has brought about significant changes, including the expansion of network sizes and their ability to support a multitude of applications. This growth has presented formidable challenges in the endeavor to secure data and network nodes effectively, especially in the face of emerging threats.

Existing IDSs have demonstrated limitations in identifying various forms of attacks, including the elusive zero-day attacks, and in reducing the false alarm rate (FAR) [3]. This underscored the pressing need for an efficient, precise, and cost-effective NIDS solution to fortify network security.

The current study introduces an innovative adaptive ensemble learning model, designed to harness the strengths of various techniques for diverse data detection scenarios. Leveraging ensemble learning, which combines predictions from multiple base estimators, this model offers enhanced generalizability and robustness compared to relying solely on a single estimator. The study utilizes the NSL-KDD dataset [4] and incorporates widely used techniques, including decision trees [5], random forests [6], K-nearest neighbors [7], and XGBoost [8], in training our model. In the realm of network security, a pivotal aspect is intrusion detection. Intrusion detection involves the process of identifying user actions that contravene an organization's established network security policies. With the escalating numbers of users, network devices, and applications, there is a growing imperative to devise innovative security measures and protective strategies to safeguard against a myriad of threats targeting internet resources. Intrusion detection systems are pivotal in monitoring network traffic to discern whether any unauthorized parties are engaged in malicious activities. The overarching aim of these systems is to effectively recognize and categorize network traffic data, particularly distinguishing malicious data from legitimate ones. In essence, the primary objective of a network intrusion detection system is the identification and classification of malicious or suspicious activities.

1.1 Objective

In the realm of network security, a pivotal aspect is intrusion detection. Intrusion detection involves the process of identifying user actions that contravene an organization's established network security policies. With the escalating numbers of users, network devices, and applications, there is a growing imperative to devise innovative security measures and protective strategies to safeguard against a myriad of threats targeting internet resources. Intrusion detection systems are pivotal in monitoring network traffic to discern whether any unauthorized parties are engaged in malicious activities. The overarching aim of these systems is to effectively recognize and categorize network traffic data, particularly distinguishing malicious data from legitimate ones. In essence, the primary objective of a network intrusion detection system is the identification and classification of malicious or suspicious activities.

1.2 Scope

Intrusion Detection Systems (IDSs) are meticulously crafted to empower organizations to remain vigilant and responsive to the ever-present threat of cyberattacks. This is achieved by gathering data from a multitude of systems and network resources, which are then scrutinized for potential security vulnerabilities. IDSs are typically deployed for the following purposes:

Continuous monitoring and in-depth analysis of user and system activity.
Comprehensive audits of system configurations and vulnerabilities.
Assessment of the integrity of vital system and data files.
Execution of statistical analyses of activity patterns, thereby enabling the identification of known attack patterns.
Detection of any abnormal or anomalous activity, with particular emphasis on monitoring operating systems.
Through the identification and alerting of suspicious network traffic, IDSs equip organizations to enhance the security of their network devices and safeguard their critical network data. In the current landscape, securing the flow of data within and between internal and external networks is of paramount importance. It is imperative to establish a resilient and adaptable intrusion detection system, particularly in the face of increasingly sophisticated and frequent cyberattacks.

The present study leverages the NSL-KDD dataset, encompassing a diverse array of network traffic flow data, and combines it with various widely adopted techniques, including decision trees, random forests, K-nearest neighbors, and XGBoost, to train our model. This research endeavor not only assesses the latest advancements and challenges in the field of intrusion detection technology but also introduces an innovative adaptive ensemble learning model. This model harnesses the power of ensemble learning to augment detection effectiveness, a critical aspect as cyberattacks continue to evolve and increase in frequency.

1.3 Organization of Report

The report is organized into five chapters. The first chapter of the report deals with the general background, objective and scope of the project. The second chapter contains various literature reviews related to the project. The third chapter contains basic information about resources. The detailed architecture and operation of the proposed system is discussed in the fourth chapter. The fifth chapter contains experimental results and discussions. The conclusions are summarized in the final chapter. In the last chapter, the future scope of the given project is also added. Finally, the links are listed on the last pages.

2. Literature Survey

Xianwei Gao and Chun Shan [9] proposed an adaptive ensemble model for intrusion detection. They used NSL-KDD dataset which is used to classify the attacks that happen in network traffic flow. It has been demonstrated that the ensemble model significantly increases the detection accuracy when compared to other research articles. The adaptive voting system they suggested has got an accuracy of 85.2 %. Shrivas [10] introduced the ANN-Bayesian Net-GR technique, which is an ensemble of Bayesian Net with Gain Ratio (GR) feature selection and Artificial Neural Network (ANN). On the KDD99 and NSL-KDD data set, they utilised a variety of individual classification approaches and its ensemble model to assess the model's robustness. With the KDD99 data set, the suggested model provides accuracy of 99.42%, and with the NSL- KDD data set, the accuracy of 98.77%, for 35 and 31 features, respectively. Ambusaidiet al.[11] implemented an algorithm based on mutual information that chooses the best feature for classification analytically. This algorithm for choosing features based on mutual information handle data features with linear and nonlinear dependencies. The features chosen by the suggested features election technique are used to construct an intrusion detection system (IDS),

known as Least Square Support Vector Machine-based IDS (LSSVM-IDS). Several intrusion detection evaluation datasets, KDD Cup 99, NSL-KDD, and Kyoto 2006+ dataset, are used to assess the performance of LSSVM-IDS.

3. Resource Background

Ensemble Learning
Ensemble learning is a powerful approach to solving computational intelligence problems. It involves the strategic development and combination of multiple models, such as classifiers or experts, in a unified ensemble learning process. The primary objective of ensemble learning is to enhance the performance of models, whether in the context of classification, prediction, feature approximation, or reducing the risk of selecting an inferior model. Additionally, ensemble learning offers various other advantages, including providing confidence levels for model inferences, selecting the best or near-best features, data fusion, incremental learning, and error correction [12].

Ensemble machine learning approaches leverage insights from different learning models to make more accurate and robust inferences. Learning models are susceptible to errors stemming from noise, variation, and bias. The use of ensemble methods in machine learning (ML) helps mitigate these error-inducing factors, contributing to enhanced accuracy and stability in ML algorithms [13]. Ensemble techniques can be categorized into several forms, with two prominent ones being:

1. **Bagging (Bootstrapped Aggregation):** Bagging, short for Bootstrapped Aggregation, involves the process of randomly selecting records from a training dataset with replacement. This concept is akin to assessing the average rating of a product based on customer reviews, rather than relying on individual user reviews. Bagging applies multiple models to the data simultaneously and averages their results to provide the final model output. This approach is effective in improving model stability and reducing variance.

2. **Boosting:** Boosting is another ensemble technique where each model's predictions are not given equal weight. Instead, if one model performs better than the others, it is assigned more weight in the decision-making process. Boosting aims to reduce bias and enhance the overall performance of the ensemble. There are various boosting techniques, including Adaboost and Gradient Boosting. Gradient Boosting, also known as Gradient Boosting Machine (GBM), is a technique that involves building decision trees with fewer leaves and scaling the tree with a learning rate. This approach is particularly useful in reducing overfitting and improving the long-term performance of models. Gradient boosting can be applied to both classification and regression tasks, and there are efficient and well-regulated open-source implementations available [22].

Ensemble learning, with its various techniques, is a valuable tool in the field of machine learning, contributing to more accurate and robust model predictions.

4. Proposed Methodology

4.1 Introduction

1. The adaptive ensemble learning model developed in this study selects well-known machine learning algorithms as alternative classifiers, including decision tree, Xgboost, kNN, and random forest. Comparative tests are used to choose four voting classifiers. The detection performance of each algorithm is then improved by altering the sampling frequency, setting data weights, multi-layer detection, and other combination methods. To get the best detection results, the adaptive voting technique is utilized with various class weights.

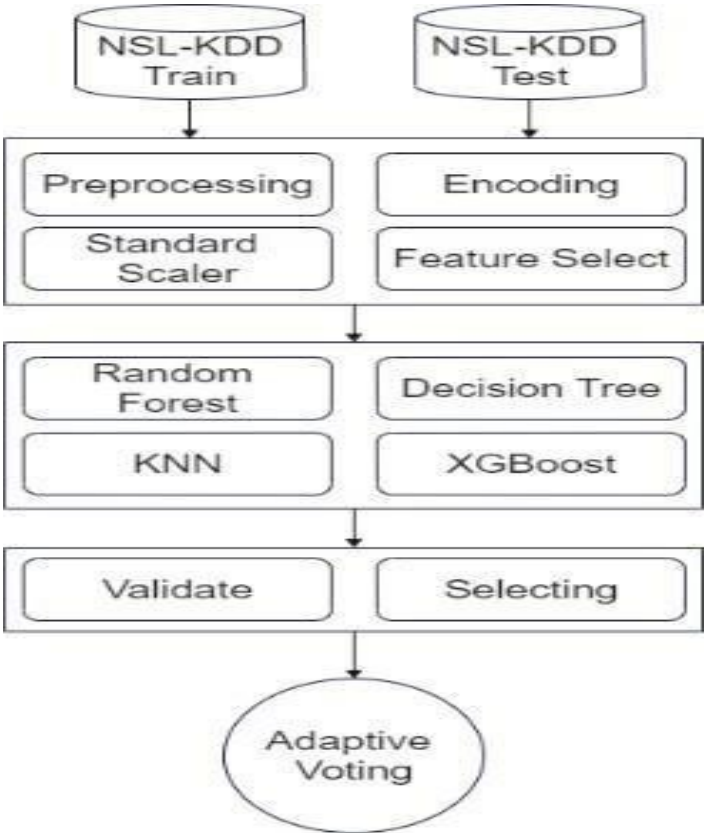

Figure 1. Adaptive ensemble model architecture

The processes listed below are primarily included in the adaptive ensemble learning model in Figure 1.

2. Input the NSL-KDD trainingdataset.
3. Input the NSL-KDD testingdataset
4. The preprocessing module converts the character-type features such as label and service into numbers, standardizes the data, and deletes unnecessaryfeatures.
5. Train each candidate algorithms by using preprocesseddata.
6. Then find accuracy of eachalgorithms
7. According to the accuracy of each algorithm, the adaptive voting algorithm model is generated.
8. Each algorithm selected is used to calculate the final voting results using the adaptive votingalgorithm.
9. Byadaptive voting, the highest possible classification test results can be calculated.

4.2 Dataset

The dataset used in this study is NSL-KDD. The NSL-KDD dataset is the refined version of one of the popular network dataset KDD cup99 [15]. Each record contains 41 attributes that describe various aspects of the flow, with labels designating each one as either an attack-type or a normal attribute.
The details of all 41 attributes found in the NSL-KDD dataset are given below.

1. Duration- Length of time duration of the connection
2. Protocol_type-Protocol used in the connection
3. Service- Destination network service used
4. Flag- Status of the connection – Normal or Error
5. Src_bytes- Number of data bytes transferred from source to destination in single connection
6. Dst_bytes- Number of data bytes transferred from destination to source in single connection
7. Land- if source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0
8. Wrong_fragment- Total number of wrong fragments in this connection
9. Urgent- Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated
10. Hot- Number of ,,hot" indicators in the content such as: entering a system directory, creating programs and executing programs
11. Num_failed _logins- Count of failed login attempts
12. Logged_in- Login Status : 1 if successfully logged in; 0 otherwise
13. Num_compromised- Number of ``compromised' ' conditions
14. Su_attempted 1 if ``su root" command attempted or used; 0 otherwise

15. Num_root- Number of ``root'' accesses or number of operations performed as a root in the connection
16. Num_file_creations- Number of file creation operations in the connection
17. Num_shells- Number of shell prompts
18. Num_access_files- Number of operations on access control files
19. Num_outbound_cmds- Number of outbound commands in an ftp session
20. Is_hot_login- 1 if the login belongs to the ``hot'' list i.e., root or admin; else 0
21. Is_guest_login- 1 if the login is a ``guest'' login; 0 otherwise
22. Count- Number of connections to the same destination host as the current connection in the past two
23. Srv_count- Number of connections to the same service (port number) as the current connection in the past two seconds
24. Serror_rate- The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (22)
25. Srv_serror_rate -The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (23)
26. Rerror_rate- The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (22)
27. Srv_rerror_rate- The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (23)
28. Same_srv_rate- The percentage of connections that were to the same service, among the connections aggregated in count (22)
29. Diff_srv_rate- The percentage of connections that were to different services, among the connections aggregated in count (22)
30. Srv_diff_host_ rate- The percentage of connections that were to different destination machines among the connections aggregated in srv_count (23)
31. Dst_host_count- Number of connections having the same destination host IP address
32. Dst_host_srv_ count Number of connections having the same port number
33. Dst_host_same_srv_rate- The percentage of connections that were to the same service, among the connections aggregated in dst_host_count (32)
34. Dst_host_diff_ srv_rate The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)

35. Dst_host_same _src_port_rate- The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_c ount (33)

36. Dst_host_srv_ diff_host_rate The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count (33)

37. Dst_host_serror_rate- The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (32)

38. Dst_host_srv_serror_rate- The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_c ount (33)

39. Dst_host_rerror_rate- The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_count (32)

40. Dst_host_srv_r error_rate- The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_c ount (33)

41. Root_shell- 1 if root shell is obtained; 0 otherwise

Data on the different 5 classes of network connection vectors, which are divided into one normal class and four attack classes, are contained in the 42nd attribute. The four attack types are further divided into DoS, Probe, R2L, and U2R categories [16]. They are:

1. DOS: Denial of service attacks prevents the victim from responding to genuine requests, such as syn flooding, by draining its resources. "Source bytes" and "percentage of packets with errors" are significantfeatures.
2. Probing: Gaining knowledge about the remote victim is the goal of surveillance and other probing attacks, such as port scanning. Relevant features include "source bytes" and "duration ofconnection."
3. U2R: Unauthorized access to local super user (root) capabilities is a form of attack in whichanattackerlogsintoavictimsystemusinganormalaccountandattemptstoget root or administrator credentials by taking advantage of a flaw in the victim, such as a buffer overflow attack. The "number of file creations" and "number of shell prompts invoked" features are significantones.
4. R2L: Unauthorized disclosure from a remote system allows the attacker to breach security and take control of the victim's machine locally. Eg: password guessing. Relevant characteristics Host-level characteristics like "number of failedlogin
5. attempts" and network level features like "duration of connection" and "service requested".

Table 1. NSL-KDD

Class	Subclass	Description
Normal	Normal	benign record
DOS	apache2,back,mailbomb,processtable,snmpgetattack,teardrop,smurf,land,Neptune,pod,udpstorm	denial-of-service, e.g. syn flood;
Probe	Nmap, Ipsweep, Portsweep, Satan, Mscan, Saint	surveillance and other probing, e.g., port scanning.
R2L	ftp_write,guess_passwd,snmpguess,imap,spy,warezclient,warezmaster,multihop,phf,imap,named,sendmail,xlock, xsnoop,worm,	unauthorized access from a remote machine, e.g. guessing password;
U2R	Ps,buffer_overflow,perl,rootkit,loadmodule,xterm,sqlattack,httptunnel,	unauthorized access to local superuser (root) privileges, e.g., various "buffer overflow" attacks;

Table 1.NSL-KDD

4.3 Data Preprocessing

The original data set contains 42 fields, among which the character types are label, protocol, and flag. In order to use those fields as input for a machine learning system, preprocessing processes are required. First, the original data's label tag columns are transformed into five types: Normal: 0, DOS: 1, Probe: 2, R2L: 3, and U2R: 4.

The values for the protocol type field are TCP, UDP, and ICMP. To process its text values, we employ one-Hot-Encoding [17], which turns all classification features into binary ones, such as [1,0,0] for the TCP protocol.

There are 122 data features left behind after transformation. The training set data analysis reveals that the value of num_outbound_cmds is zero, hence this feature is eliminated. Many feature fields in the original data have a wide range of values, which has a significant impact on the training results. As a

result, the data is standardized using the Standard Scaler [18] approach. Standardized data is changed by taking away the mean and dividing it by the variance (or standard deviation). The normalised data follows the conventional normal distribution, which has a mean of 0 and a standard deviation of 1. An algorithm for machine learning can be used to train once the training data set has been preprocessed. The test set utilizes the same pre-processing metrics as the training set and the data from the NSL-KDD test dataset. The classification algorithm should work to increase the accuracy of small ratio data in addition to considering the overall detection accuracy.

4.4 Classification Algorithms

The features that are selected can be given as input to the classifiers. Here the classification models used are Decision Tree, RF, XGBoost and KNN. Here these models can be trained by using the features that are selected before. After that with the test dataset the model tries to predict the attack classes. Finally by using an adaptive voting classifier an ensemble model is created, this will helps to find the best accuracy among all the classifiers and also get the overall accuracy.

In the context of network intrusion detection, several classification models are employed to predict attack classes. The selected classification models for this study are:

1. **Decision Tree:** Decision trees are highly effective and popular tools for categorization and prediction. They take the form of a tree structure that resembles a flowchart, with internal nodes representing attribute tests, branches representing test results, and leaf nodes denoting class labels. Decision trees are known for their ability to produce clear and interpretable rules. They are computationally efficient and excel in classification tasks.
2. **Random Forest (RF):** Random Forest is a widely used ensemble learning technique for classification and regression problems. It builds multiple decision trees on different subsets of the data and combines their predictions through majority voting. This approach helps improve accuracy and reduces the risk of overfitting as the number of trees in the forest increases. The class that the majority of trees choose becomes the output of the random forest for classification problems.
3. **XGBoost:** eXtreme Gradient Boosting, often abbreviated as XGBoost, is a machine learning algorithm based on decision trees and a gradient boosting framework. XGBoost involves the sequential creation and addition of decision trees, with each tree attempting to correct the mistakes made by its predecessors. This algorithm is renowned for its exceptional accuracy and optimization for speed, making it a popular choice in machine learning.
4. **K-Nearest Neighbors (KNN):** K-Nearest Neighbors is a straightforward and intuitive classification method. It classifies a data point based on the majority class of its K-nearest neighbors in feature space. The local estimation provided by KNN does not rely on the global distribution, making it suitable for cases where a global probability distribution is not easily determined.
5. **Voting Classifier:** A voting classifier is an ensemble model that combines the outputs of multiple individual classifiers. It trains on a collection of different models and makes predictions based on the class with the highest likelihood among the models. This approach leverages the wisdom of multiple models to improve overall prediction accuracy.

After training these classification models with the selected features, they are tested using the test dataset to predict attack classes. To further enhance the prediction accuracy, an adaptive voting classifier is employed. The ensemble model combines the outputs of the individual classifiers, aiding in achieving the best accuracy among all the classifiers and providing an overall accuracy score

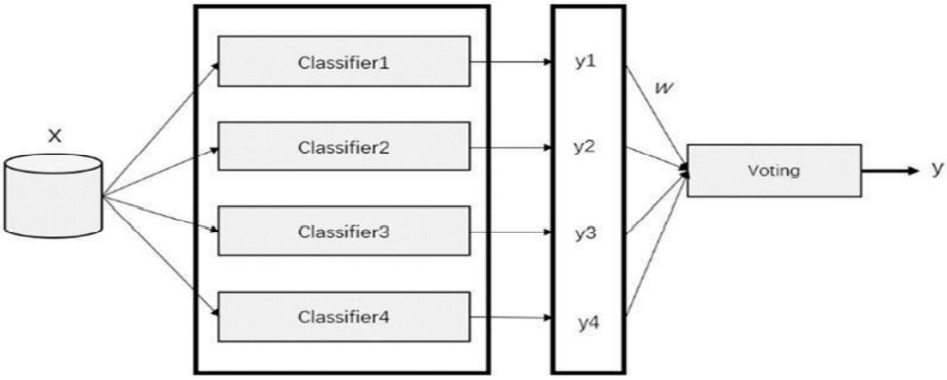

Figure 2. Adaptive Voting algorithm

Figure 2. Adaptive Voting algorithm is used to predict the output class based on the highest majority of votes, it merely averages the results of each classifier that was passed into the voting classifier. The concept is to build a single model that learns from these models and predicts output based on their aggregate majority of voting for each output class, rather than building separate dedicated models and determining the accuracy for each of them. Two types of voting classifiersare:

Hard Voting: The class with the highest majority of votes, or the class that had the highest likelihood of being predicted by each of the classifiers, is the predicted output class in hard voting. In this case, the majority predicted A as the output when three classifiers (A, A, and B) predicted the output class. Therefore, the final prediction will beA.

Soft Voting: In soft voting, the prediction mode for each output class is based on the averageprobabilityassignedtothatclass.Assumethatgivensomeinput,theprediction probabilities for classes A and B are (0.30, 0.47, and 0.53) and (0.20, 0.32, 0.40). As a result, class A's average is 0.4333, while class B's average is 0.3067. As a result, class A is the winner because it had the highest probability as averaged by allclassifiers.

4.5 Evaluation Matrix

The classification results are evaluated by using the accuracy of different models

To train our models, we will use NSL-KDD dataset. So, for our study, we will use the following metrics: precision, recall and F1-score [20]:
Precision (P): is calculated by dividing the number of true positive predictions (Tp) by the total number of true positive predictions (Tp) plus the number of false positives (Fp).

$$P = \frac{Tp}{Tp+Fp} \qquad (1)$$

Recall (R): The number of true positives (Tp) divided by the total number of true positives plus false negatives (Fn).

$$R = \frac{Tp}{Tp+} \qquad (2)$$

F1 Score: is the weighted average of precision and recall.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3)$$

When performing a K-FOLD Cross Validation [24], the following steps are taken:

- Divide the data to k equalparts (folds)
- One fold is used for testing while the others are used in training The process is repeated for all the folds and the accuracy is averaged.

5. Experiment Results

The classifiers used in this work to classify the attack classes are Decision tree, RF, KNN and XGBoost. The accuracy of random forest and xgboost classifiers was the highest among others. The accuracy got when ensemble the model is 83.63% for Normal, 94.41% for DOS, 99.15% for Probe, 88.63% for R2L and 99.55% for U2R. The mean accuracy [21] of all the attack classes is 92.59%.

Table 2 .Comparative Evaluation of different classifiers:

	Random Forest	KNN	Decision Tree	XGBoost	Ensemble
Attack Type	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
Normal	94.12	47.259	68.231	96.46	83.632
DOS	99.38	56.747	89.22	99.26	94.415
Probe	99.05	80.045	96.513	98.755	99.151
R2L	95.52	77.096	87.353	96.634	88.639
U2R	99.62	99.315	99.56	99.632	99.55
					92.59%(Mean)

Table 2 .Comparative Evaluation of different classifiers

We must figure out how to improve the over all detection effect if we want to. Although RF and XGBoost have a superior overall detection effect. While different categorization algorithms may not all have advantages for all sorts of data, each offers advantages of its own. The total detection effect will be enhanced in the future by optimizing the combination of algorithms and utilizing the benefits of multiple methods

Screenshots: Working Model Demo

Figure 3.Accuracy of RF

Figure 3 which show the accuracy of RF Model working in Jupiter notebook which had the accuracy of 99.6%

```
[34] accuracy = cross_val_score(clf_KNN_Normal, X_Normal_test2, Y_Normal_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.47259

[35] accuracy = cross_val_score(clf_KNN_DoS, X_DoS_test2, Y_DoS_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.56747

[36] accuracy = cross_val_score(clf_KNN_Probe, X_Probe_test2, Y_Probe_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.80045

[37] accuracy = cross_val_score(clf_KNN_R2L, X_R2L_test2, Y_R2L_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.77096

[38] accuracy = cross_val_score(clf_KNN_U2R, X_U2R_test2, Y_U2R_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.99315
```

Figure 4.Accuracy of KNN

Figure 4 which show the accuracy of KNN Model working in Jupiter notebook which had the accuracy of 99.3%

```
[41] accuracy = cross_val_score(clf_dt_rfeNormal, X_Normal_test2, Y_Normal_test, cv=5, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.68231

     accuracy = cross_val_score(clf_dt_rfeDoS, X_DoS_test2, Y_DoS_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.89220

[43] accuracy = cross_val_score(clf_dt_rfeProbe, X_Probe_test2, Y_Probe_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.96513

[44] accuracy = cross_val_score(clf_dt_rfeR2L, X_R2L_test2, Y_R2L_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.87353

[45] accuracy = cross_val_score(clf_dt_rfeU2R, X_U2R_test2, Y_U2R_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.99560
```

Figure 5.Accuracy of Decision tree

Figure 5 which show the accuracy of Decision tree Model working in Jupiter notebook which had the accuracy of 99.5%

```
[48] accuracy = cross_val_score(clf_xgb_Normal, X_Normal_test2, Y_Normal_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.96460

[49] accuracy = cross_val_score(clf_xgb_DoS, X_DoS_test2, Y_DoS_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.99260

[50] accuracy = cross_val_score(clf_xgb_Probe, X_Probe_test2, Y_Probe_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.98755

[51] accuracy = cross_val_score(clf_xgb_R2L, X_R2L_test2, Y_R2L_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.96634

[52] accuracy = cross_val_score(clf_xgb_U2R, X_U2R_test2, Y_U2R_test, cv=10, scoring='accuracy')
     print("Accuracy: %0.5f" % (accuracy.mean()))

     Accuracy: 0.99632
```

Figure 6. Accuracy of Xgboost

Figure 6which show the accuracy ofXgbooster Model working in Jupiter notebook which had the accuracy of 99.6

```
[ ] print("Accuracy: %0.5f" % (accuracy.mean()))

    Accuracy: 0.83668

[ ] accuracy = cross_val_score(clf_voting_DoS, X_DoS_test2, Y_DoS_test, cv=10, scoring='accuracy')
    print("Accuracy: %0.5f" % (accuracy.mean()))

    Accuracy: 0.94397

[ ] accuracy = cross_val_score(clf_voting_Probe, X_Probe_test2, Y_Probe_test, cv=10, scoring='accuracy')
    print("Accuracy: %0.5f" % (accuracy.mean()))

    Accuracy: 0.99134

[ ] accuracy = cross_val_score(clf_voting_R2L, X_R2L_test2, Y_R2L_test, cv=10, scoring='accuracy')
    print("Accuracy: %0.5f" % (accuracy.mean()))

    Accuracy: 0.88671

[ ] accuracy = cross_val_score(clf_voting_U2R, X_U2R_test2, Y_U2R_test, cv=10, scoring='accuracy')
    print("Accuracy: %0.5f" % (accuracy.mean()))

    Accuracy: 0.99550

[ ] print("Mean of Whole Accuracy = ")
    print(((0.99 + 0.88 + 0.99 + 0.94 + 0.83)/5))

    Mean of Whole Accuracy =
    0.9259999999999999
```

Figure 7 Final accuracy of Ensemble

Figure 7 which show the accuracy of Ensemble Model working in Jupiter notebook which had the accuracy of 92.9

6. Conclusion and Future Scope

This work provides comprehensive network intrusion detection systems to classify different attack classes according to its characteristic features by using ensemble learning. No learningalgorithmisthebestlearnerinanysituation,accordingtotheideathatthereisnosuch thing as a free lunch. The performance differences between each algorithm are not significant in the detection effect of a single classification technique. Whatever learning algorithm is chosen, a number of techniques can be applied to improve the effectiveness of detection.

Here, I proposed an approach for adaptable ensemble learning. The model's main concept is to employ ensemble learning to aggregate the benefits of various algorithms. To enhance the detection effect, I employ the ensemble learning technique. It has been demonstrated that this ensemble model significantly increases the detection accuracy when compared to other research articles. The accuracy of the adaptive voting algorithm I proposed is 92.59%. Although RF and XGBoost have a superior overall detection effect. The main objective of the subsequent practical systems in the intrusion detection field is to maximize feature extraction and preprocessing techniques, enhance training data quality, and increase the data's separability. In the area of network security research and application, ensemble machine learning offers a good generalisation effect that merits ongoing promotion and optimization. The proposed classification technique is for determining various attack classes. Thisnecessitatesthenetworkdatabelongingtoorbeingavariantofknownattackclasses.The scope of this project does not include the identification of live network attacks types. In the future, we can try together real-time network data and categorize different attack types.

Reference

[1] C. C. Aggarwal, “Outlier Analysis,” in Data Mining: The Textbook. Cham: Springer International Publishing, 2015, pp.237–263.

[2] O. Akinrolabu, I. Agrafiotis, and A. Erola, “The challenge of detecting sophisticated attacks: Insights from soc analysts,” in Proceedings of the 13th International Conference on Availability,Reliability and Security, ser. ARES 2018. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https: //doi.org/10.1145/3230833.3233280

[3] D. Bruns-Smith, M. M. Baskaran, J.Ezick, T. Henretty, and R. Lethin, “Cyber security through multidimensional data decompositions,” in 2016 Cybersecurity Symposium (CYBERSEC), 2016, pp.59–67.

[4] P. Casas, F. Soro, J. Vanerio, G. Settanni,and A. D’Alconzo, “Network security and anomalydetection with big-dama, a big data analytics framework,” in

[5] 2017 IEEE 6th International Conference on Cloud Networking (CloudNet), 2017, pp. 1–7.

[6] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Comput. Surv., vol. 41, 07 2009.

[7] V.Chang,Y.-H.Kuo,andM.Ramachandran,“Cloudcomputingadoptionframework:Asecurityframework for business clouds,” Future GenerationComputer Systems, vol. 57, pp. 24–41, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X15003118

[8] V. Chang and M. Ramachandran, “Towardsachieving data security with the cloud computing adoption framework,” IEEE Transactions on Services Computing, vol. 9, no. 1, pp. 138–151,2016.

[9] A.A.Ca´rdenas,P.K.Manadhata,andS.P.Rajan,“Bigdataanalyticsforsecurity,”IEEESecurityPrivacy, vol. 11, no. 6, pp. 74–76,2013.

[10] M.Fiedler,“Cyberangriffe:dramatischezunahmeund rekordscha¨den,” procontra-online.de, AlsterspreeVerlag GmbH, Berlin, 11 2019, accessed:2020-04-23.

[11] R. F. Fouladi, O. Ermis¸, and E. Anarim, "A ddos attack detection and defense scheme using time- series analysis for sdn," Journal of Information Security and Applications, vol. 54, p. 102587, 2020. [Online]. Available:https://www.sciencedirect.com/science/article/pii/S2214212620307560

[12] "Glacier (angriffserkennung durchmultidimensionaleanalysesicherheitsrelevanter datenstro¨me) projecthomepage," https://www.glacier-project.de/index.php/home-eng.html, accessed: 2020-04-14.

[13] M. Gupta, J. Gao, C. C. Aggarwal, and J.Han,"Outlierdetectionfortemporaldata," in Synthesis Lectures on Data Mining and Knowledge Discovery, Vol. 5, No. 1, 3 2014, pp.1–129.

[14] F. Heine, C. Kleiner, P. Klostermeyer, V.Ahlers, T. Laue, and N. Wellermann, "Detecting attacks in network traffic using normality models: The cellwise estimator," in Proceedings of the 14th International Symposium on Foundations & Practice of Security, Dec 2021, p. TBD. [Online]. Available: TBD

[15] M. A. Jabbar, R. Aluvalu, and S. S. S. Reddy, "Cluster based ensemble classification for intrusiondetection system," in Proceedings of the 9th International Conference on Machine Learning and Computing, ser. ICMLC 2017. New York, NY, USA: Association for Computing Machinery,2017, p.253–257.

[16] S. Khaliq, Z. U. Abideen Tariq, and A. Masood, "Role of user and entity behavior analytics in detecting insider attacks," in 2020International

.