Proceedings of 14th IOE Graduate Conference
Peer Reviewed
Year: 2023 Month: December Volume: 14
ISSN: 2350-8914 (Online), 2350-8906 (Print)

Intrusion Detection System for Datacenter Network using Ensemble Learning

Subarna Bastakoti [a], Sharad Kumar Ghimire [b]

[a, b] *Department of Electronics and Computer Engineering, Pulchok Campus, IOE, Tribhuvan University, Nepal*
[a] 078msice018.subarna@pcampus.edu.np, [b] skghimire@ioe.edu.np

Abstract

Network intrusion detection systems (NIDS) play a critical role in safeguarding the integrity and security of network infrastructures. However, the increasing complexity and dynamic nature of data centre network environment have presented significant challenges to the effective detection and mitigation of network intrusions. This work aims to address this issue by proposing a novel approach to NIDS specifically designed for network environment, utilizing ensemble learning techniques. The proposed NIDS leverages the power of ensemble learning, which combines multiple detection algorithms to enhance the overall accuracy and robustness of intrusion detection. Multiple learning algorithms, are integrated into the NIDS framework to effectively capture a wide range of intrusion behaviors, enabling the NIDS to adapt and learn from evolving attack techniques. To evaluate the performance of the proposed NIDS, a comprehensive simulation and experimentation framework is established using powerful GNS3 platform and data sets made from real world network intrusion traffic (UNSW-NB15 data set).The evaluation process involves various performance metrics, including accuracy, precision, recall, and F1-score, to assess the effectiveness and efficiency of the proposed NIDS. The outcomes of this research include a robust and adaptive NIDS tailored specifically for network environment. The ensemble learning has the potential to improve the accuracy of intrusion detection and reduce false positive rates, thereby enhancing the security posture of network based systems.

Keywords

Network Intrusion Detection System, GNS3, NIDS, Ensemble Learning, Network Security, Intrusion Detection

1. Introduction

With the widespread adoption of data networks, organizations are increasingly relying on network-based infrastructures to store and process their critical data [1]. While network computing offers numerous benefits, such as scalability, cost-efficiency, and flexibility, it also introduces new security challenges [2]. Network intrusions in computing environments can have severe consequences, including unauthorized access, data breaches, service disruptions, and potential financial losses. Therefore, the development of effective network intrusion detection systems (NIDS) specifically designed for network infrastructure is of paramount importance [3]. Traditional NIDS solutions often struggle to cope with the dynamic and complex nature of current network environments [4]. The inherent characteristics of network-based systems, such as shared resources, virtualization, and the high volume of network traffic, pose unique challenges for intrusion detection [5]. Consequently, there is a pressing need for innovative approaches that can adapt to the evolving threat landscape and provide robust security measures [6].

Recent statistical data reveal that a large number of security breaches occur in the virtual network layer of network [7]. The existing/traditional intrusion detection systems (IDS) are turning obsolete because of huge network traffic and its dynamic behavior [8]. The increase in computational power and resources has largely complicated the ways in which these network attacks could be launched [9]. Many researches in this domain are converging towards machine learning approach, hence an ensemble learning based IDS could be a solution to efficient Network intrusion detection [10]. ML based detection approaches are capable of dealing with evolving and newer threat types compared to existing IDS techniques [11].

2. Literature Review

Contemporary research direction is focused on using a standalone machine learning or deep learning algorithm for network intrusion detection. The motivation behind this research stems from the inadequacies of existing NIDS solutions in network infrastructure. While ensemble learning techniques have shown promise in improving the accuracy and efficiency of intrusion detection, their potential in the context of network computing remains largely unexplored. By using ensemble learning, it is possible to harness the complementary strengths of different detection algorithms and enhance the overall effectiveness of intrusion detection. Further results from contemporary literature show that most of the researches have been conducted with dated data set (NSL-KDD), which can represent the standard network intrusion types but cannot wholly include the newer intrusion types. Hence, this work proposes a ensemble based NIDS capable of detecting evolving kind of intrusions in network environment.

Krishnaveni et al. [1] developed an efficient IDS for cloud environment using ensemble feature selection and classification technique using the NSL-KDD, Kyoto and Honeynet datasets. The ensemble based classification method classified whether the network traffic behavior is normal or attack. They proposed the univariate ensemble filter feature selection (UEFFS) method using five filters to get the optimal feature subsets from the considered dataset.

Ahmad et al. [5] believes in a comprehensive assessment of recent articles focusing on NIDS, examining the strengths and limitations of the proposed solutions and explores current trends and advancements in ML and DL-based NIDS, considering aspects such as the methodology employed, evaluation metrics used, and dataset selection.

Li et al. [7] discusses about two groups of IDS; one by fuzzy logic and other by artificial neural network (ANN) using the KDD99 dataset in data mining environment suggesting that the increasing volume and velocity of generated data require new and improved ML based NIDS.

Yang et al. [8] did a systematic literature review of contemporary literature on NIDS on 119 top cited papers investigating the technical domain of the research including application, data preprossessing, attack-detection techniques, datasets and evaluation parameters.

Pham et al. [10] proposed the use of ensemble models in Intrusion Detection Systems (IDS) to improve their accuracy and reduce false alarm rates. The authors proposed two ensemble techniques, Bagging and Boosting, and apply them to the NSL-KDD dataset. They also use two different feature selection techniques to select the best features for the ensemble models namely the J48 as the base classifier and the 35-feature subset.

Yin et al. [12] proposed a NIDS (Network Intrusion Detection System) based on an RNN (Recurrent Neural Network) model. The model exhibits higher complexity and necessitates additional training time. The outcomes indicate reduced detection rates specifically for less common attack categories.

A NIDS (Network Intrusion Detection System) model is proposed by Ali et al. [13] utilizing FLN (Fast Learning Network) and the particle swarm optimization algorithm. The performance of the proposed model surpasses that of other FLN-based models employing various optimization algorithms. Limited training data caused low detection rate in this study.

A NIDS is proposed by Jia et al. [14] employing a DNN (Deep Neural Network) with four hidden layers, demonstrating superior performance compared to IDS (Intrusion Detection System) based on traditional ML (Machine Learning) methods.

Jiang et al. [15] proposed an IDS (Intrusion Detection System) incorporating CNN (Convolutional Neural Network) and BiLSTM (bi-directional long short-term memory) in a deep hierarchical structure with SMOTE (Synthetic Minority Over-sampling Technique) to increase the representation of minority samples using both older (NSL-KDD) and newer (UNSW-NB15) datasets for comprehensive assessment.

The study by Karatas et al. [16] provides an analysis of six ML-based NIDSs that tackle dataset imbalance by mitigating the imbalance ratio through the use of SMOTE. The detection rate for the minority attack class is enhanced. Higher accuracy was achieved with the use of Adaboost algorithm using the latest CSECIC-IDS2018 dataset.

Malaiya et al. [17] focused on various models for ID (Intrusion Detection), employing fully connected networks, Variational Autoencoders (AE), and Seq2Seq structures, respectively.Among the different models proposed, the Seq2Seq model exhibits the highest performance, with increased training overhead compared to the others.

Zhang et al. [18] suggested a multilayer NIDS model incorporating CNN (Convolutional Neural Network) and gcForest algorithms with a novel P-Zigzag algorithm for converting raw data into a two-dimensional greyscale image evaluated with the combination of the UNSW-NB15 and CIC-IDS2017 datasets.

Wei et al. [19] worked on DL (Deep Learning)-based model, known as DBN (Deep Belief Network), and proposed and optimized through a combination of various optimization algorithms, including Particle Swarm, Fish Swarm, and Genetic Algorithm.The proposed model exhibits complexity and necessitates a longer training time.

Gao et al. [20] believed in an adaptive ensemble model, incorporating several base classifiers such as decision tree, random forest (RF), K-Nearest Neighbor (KNN), and DNN facilitating best classifier selection through an adaptive voting algorithm using the NSL-KDD dataset.

Ravi et al. [21] studied to monitor effective real-time monitoring of network traffic for intrusion detection using a hybrid scalable DNN (Deep Neural Network) framework called scale-hybrid-IDS-AlertNet giving higher detection rates on majority class attacks.

Yao et al.[22] used a multilevel intrusion detection model by integrating a clustering concept with RF (Random Forest) demonstrating superior performance in detecting attacks with lower instances using the KDD Cup'99 dataset.

Khan et al. [23] employed an efficient two-stage NIDS model utilizing deep stacked AE (Autoencoder). In the first stage, the output is combined with probability scores and used as an additional feature in the final decision stage, enhancing classification for both normal and multi-class attacks.

A self-taught learning and MAPE-K framework are combined to propose an autonomous misuse detection system in Papamartzivanos et al. [24] to learn relevant and valuable features with a sparse AE (Autoencoder) system and model validation using the KDD Cup'99 and NSLKDD data sets.

Ahmad et al. [25] suggested an efficient network intrusion detection and classification system based on AdaBoost-based approach for network intrusion detection using UNSW-NB 15 dataset for network anomaly detection and employing SVM and MLP for comparison.

3. Methodology

3.1 IDS Implementation on Network Environment

The block diagram of the proposed IDS implementation on *GNS3* network environment is shown in Figure 1. For custom network traffic generation; a python api called *scapy* was used that generated various kinds of network traffic(normal and intrusion traffic) the generated traffic was sniffed using the python module *NPCAP* and with the help of *Wireshark* application. The resulting packet capture file was *pcap* file that contained individual packets characteristics/features. The resulting *pcap* file was further processed in *wireshark* to finally create a *CSV* file, that was fed to the trained and saved ensemble NIDS model to obtain decision on the generated custom/real-time network traffic.

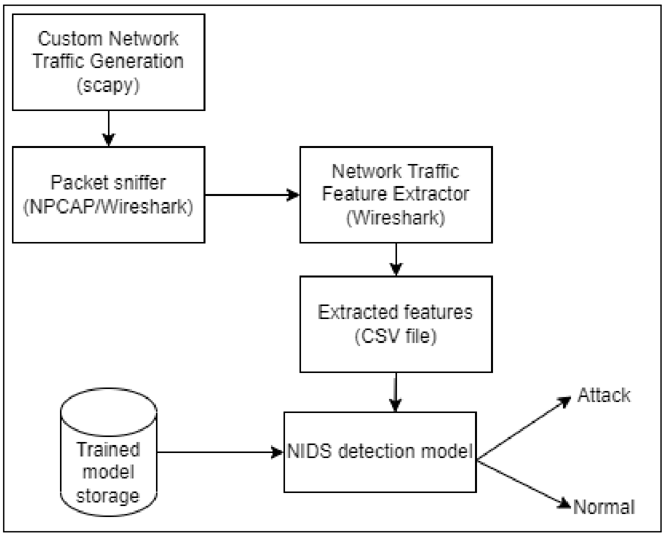

Custom Network Traffic Generation (scapy)
Packet sniffer (NPCAP/Wireshark)
Network Traffic Feature Extractor (Wireshark)
Extracted features (CSV file)
Trained model storage
NIDS detection model
Attack
Normal

Figure 1: IDS implementation on network environment

3.2 Model Training and Testing

Several experiments were performed for training and testing the ensemble NIDS model on standard UNSW-NB15 data set; further experiments on test network creation, custom data set creation, saving trained model and predicting, validating and evaluating the output of the custom data set were carried out. First the UNSW-NB15 data set was separated into training set and testing set, then the rule definition to filter the data set is done, for the case of training purpose a rule was defined to include the data satisfying *(sttl less than or equal to 61.00 and ct state ttl less than 2.00) or (sttl greater than 61.00)*. The rule was formulated by performing *dtreeviz* function on the data set. This rule definition filtered 12 percent of data and hence balancing the data set. Then binary classification of the UNSW-NB data set was done using Random Forest, KNN, Naive Bayes, Logistic Regression and XGBoost base classifiers.

The classification results were compared in terms of precision, recall and accuracy for result validation and evaluation.

In the ensemble learning block, most important features that impacts multi-class classification was found out using *feature imp = feature imp.sort values ('Importance',ascending=False) .reset index(drop=True)*. It was found out that top five features that impact the classification were *'dload', 'rate', 'sload', 'dur' and 'sttl'*. Considering those five important features the original data set was transformed in such a way that it matches the custom data set generated during our experiment in *GNS3*. A multi-class ensemble model was trained using five base classifier algorithm mentioned above and stored in local storage using *pickle* python library.

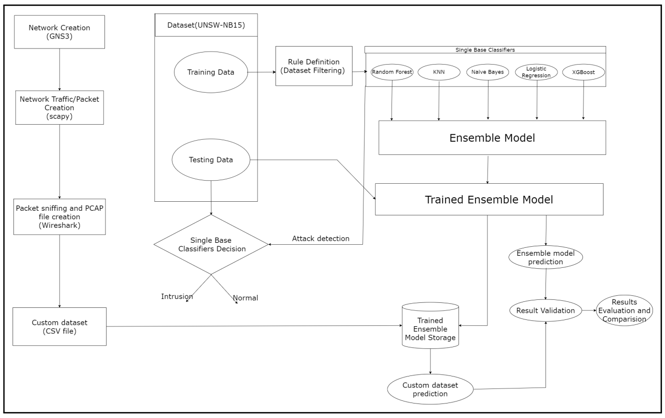

Network Creation (GNS3)
Network Traffic/Packet Creation (scapy)
Packet sniffing and PCAP file creation (Wireshark)
Custom dataset (CSV file)
Dataset(UNSW-NB15)
Training Data
Testing Data
Rule Definition (Dataset Filtering)
Single Base Classifiers
Random Forest
KNN
Naive Bayes
Logistic Regression
XGBoost
Ensemble Model
Trained Ensemble Model
Attack detection
Single Base Classifiers Decision
Intrusion
Normal
Ensemble model prediction
Result Validation
Results Evaluation and Comparision
Trained Ensemble Model Storage
Custom dataset prediction

Figure 2: Block diagram of the proposed Intrusion detection model

3.3 Test Network and custom testing data creation

The next part of the experiment was creating a test network in *GNS3*. The network contained two *cisco* router's IOS and a host connected to the the local computer through a virtual network adapter called the *Loopbacktest* interface. Custom generated intrusion traffic and normal traffic(IP based packets) were fed to the test network with the help of *python scapy module*, and packet capturing/sniffing was done using *wireshark* application.

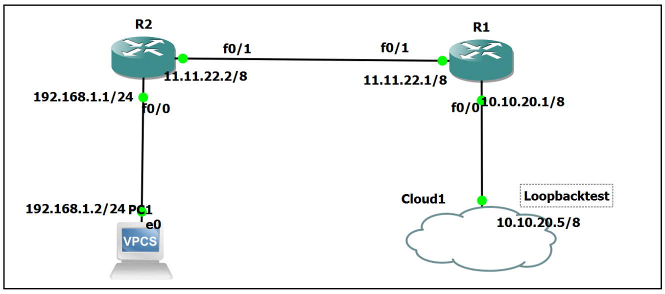


Figure 3: Test Network diagram

The captured traffic resulted in pcap file which on further processing produced a CSV file that served as the testing file in the previously trained ensemble model that was stored in the local storage. Then result validation and evaluation was done using parameters like Confusion matrix, Accuracy, Precision, Recall, and F1 score.

3.4 Instrumentation Required

The network simulation part was carried out in GNS3 network emulator/simulator with the help of python api and scapy library. The network traffic sniffing/capturing was done using NPCAP module and wireshark application, similarly the resulting *pcap file* was processed in wireshark to finally obtain the csv file that served as our custom data set in the experiment. The individual base classifier training and ensemble model building was done using Python programming language and machine learning library Scikit-learn on Intel CoreTM i7 processor with 32 GB RAM and 8 GB graphics card. Python programming was used as it includes a large set of libraries and frameworks for machine learning development. The Scikit-learn library is considered a robust and most useful library for machine learning which is used to provide building block for the models.

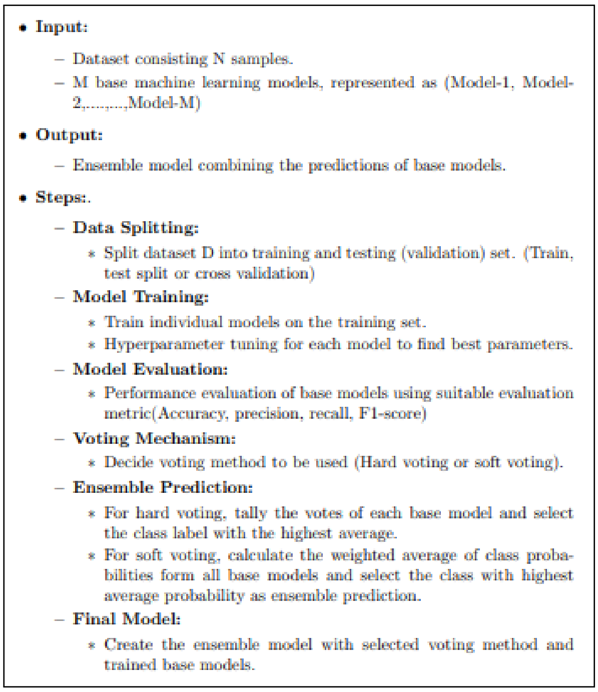
3.5 Ensemble Learning

The ensemble methods in machine learning combined the insights obtained from multiple learning models to facilitate accurate and improved decisions. Ensemble learning refers to a technique in which several models, such as classifiers or experts, are created and merged together to address a specific computational intelligence problem. Its primary goal is to enhance the performance of a model in various areas, such as classification, prediction, and function approximation, by avoiding the selection of an under-performing model [1]. Additionally, ensemble learning can be applied to determine the confidence level of a model's decision, select the best features, fuse data, support incremental learning, adapt to changing data patterns, and rectify errors.

Voting ensemble starts by building two or more independent models using your training data. Afterwards, we can use a Voting Classifier to bundle our models together and average the predictions of each sub-model when making predictions for new data [20].

The base classifiers considered for the ensemble algorithm are: Random Forest(RF), K-Nearest Neighbour(KNN), Naive Bayes (NB), Logistic Regression(LR) and GradientvBoosting(XGB).

3.5.1 Ensemble Voting Algorithm



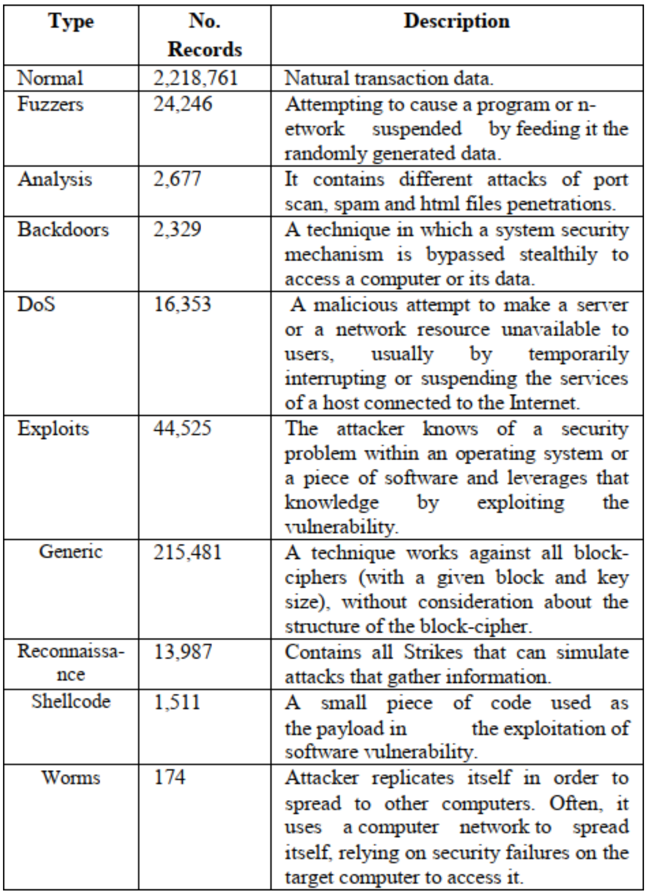
- **Input:**
  - Dataset consisting N samples.
  - M base machine learning models, represented as (Model-1, Model-2,.....,...,Model-M)
- **Output:**
  - Ensemble model combining the predictions of base models.
- **Steps:**.
  - **Data Splitting:**
    - Split dataset D into training and testing (validation) set. (Train, test split or cross validation)
  - **Model Training:**
    - Train individual models on the training set.
    - Hyperparameter tuning for each model to find best parameters.
  - **Model Evaluation:**
    - Performance evaluation of base models using suitable evaluation metric(Accuracy, precision, recall, F1-score)
  - **Voting Mechanism:**
    - Decide voting method to be used (Hard voting or soft voting).
  - **Ensemble Prediction:**
    - For hard voting, tally the votes of each base model and select the class label with the highest average.
    - For soft voting, calculate the weighted average of class probabilities form all base models and select the class with highest average probability as ensemble prediction.
  - **Final Model:**
    - Create the ensemble model with selected voting method and trained base models.

Figure 4: Ensemble Voting algorithm [10]

3.6 Data set Expalanation

The data set used for training the ensemble based Network Intrusion Detection model is UNSW-NB15. This is comparatively newer data set compared to popular NSL-KDD intrusion data set and also includes newer types of intrusion data [23]. The UNSW-NB15 dataset is a widely used network security dataset for intrusion detection system (IDS) research and evaluation [23]. It was created by the University of New South Wales (UNSW) in Australia to address the limitations of existing datasets, such as the NSL-KDD dataset.

The UNSW-NB15 dataset comprises network traffic data collected from a real-world environment, including both normal and malicious instances. It covers a wide range of attack types, such as denial of service (DoS), reconnaissance, exploitation, and infiltration. The data set contains various features related to network protocols, source and destination IP addresses, service types, and more [25].



Type	No. Records	Description
Normal	2,218,761	Natural transaction data.
Fuzzers	24,246	Attempting to cause a program or n-etwork suspended by feeding it the randomly generated data.
Analysis	2,677	It contains different attacks of port scan, spam and html files penetrations.
Backdoors	2,329	A technique in which a system security mechanism is bypassed stealthily to access a computer or its data.
DoS	16,353	A malicious attempt to make a server or a network resource unavailable to users, usually by temporarily interrupting or suspending the services of a host connected to the Internet.
Exploits	44,525	The attacker knows of a security problem within an operating system or a piece of software and leverages that knowledge by exploiting the vulnerability.
Generic	215,481	A technique works against all block-ciphers (with a given block and key size), without consideration about the structure of the block-cipher.
Reconnaissa-nce	13,987	Contains all Strikes that can simulate attacks that gather information.
Shellcode	1,511	A small piece of code used as the payload in the exploitation of software vulnerability.
Worms	174	Attacker replicates itself in order to spread to other computers. Often, it uses a computer network to spread itself, relying on security failures on the target computer to access it.

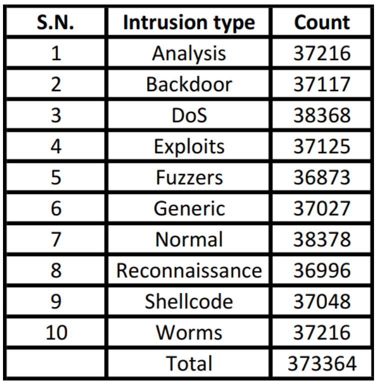
Table 1: Intrusion types and count in UNSW-NB15 intrusion dataset [26]

4. Results

4.1 Dataset Creation

In the test network created in GNS3, intrusion and normal traffic were fed with the help of *scapy* module *via* the virtual *Loopbacktest* network adapter that is connected to the test network. Two different domains in the ip pool of *10.10.20.1/8* and *11.11.22.1/8* are created where the former ip range represents the source and the bridge between source and the target test network, the later ip range represents the destination or the target test network. Upon feeding various intrusion traffic the following results were obtained.

A total of 373364 data instances were created(ip packetes) with 13 features. The features were extracted *via* the packet capture file(pcap file) made by processing the packets captured in *wireshark* over multiple sessions. The extracted features was then mapped to match the features from standard data set such that training/testing of the ensemble model on standard data set(UNSW-NB15) and testing on the custom data set could be done.

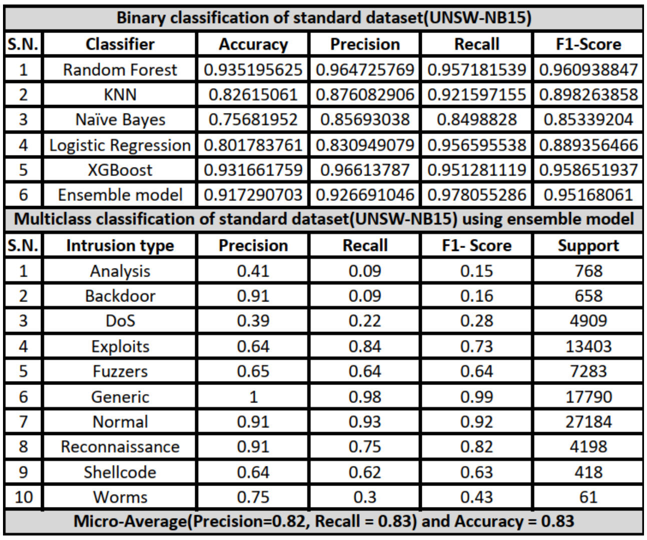


S.N.	Intrusion type	Count
1	Analysis	37216
2	Backdoor	37117
3	DoS	38368
4	Exploits	37125
5	Fuzzers	36873
6	Generic	37027
7	Normal	38378
8	Reconnaissance	36996
9	Shellcode	37048
10	Worms	37216
	Total	373364

Table 2: Custom dataset intrusion and count

4.2 Model Training and Classification on standard dataset

Binary and multi-class ensemble model was trained using the five base classifiers, the following results were obtained.



<table>
<tr><th colspan="6">Binary classification of standard dataset(UNSW-NB15)</th></tr>
<tr><th>S.N.</th><th>Classifier</th><th>Accuracy</th><th>Precision</th><th>Recall</th><th>F1-Score</th></tr>
<tr><td>1</td><td>Random Forest</td><td>0.935195625</td><td>0.964725769</td><td>0.957181539</td><td>0.960938847</td></tr>
<tr><td>2</td><td>KNN</td><td>0.82615061</td><td>0.876082906</td><td>0.921597155</td><td>0.898263858</td></tr>
<tr><td>3</td><td>Naïve Bayes</td><td>0.75681952</td><td>0.85693038</td><td>0.8498828</td><td>0.85339204</td></tr>
<tr><td>4</td><td>Logistic Regression</td><td>0.801783761</td><td>0.830949079</td><td>0.956595538</td><td>0.889356466</td></tr>
<tr><td>5</td><td>XGBoost</td><td>0.931661759</td><td>0.96613787</td><td>0.951281119</td><td>0.958651937</td></tr>
<tr><td>6</td><td>Ensemble model</td><td>0.917290703</td><td>0.926691046</td><td>0.978055286</td><td>0.95168061</td></tr>
<tr><th colspan="6">Multiclass classification of standard dataset(UNSW-NB15) using ensemble model</th></tr>
<tr><th>S.N.</th><th>Intrusion type</th><th>Precision</th><th>Recall</th><th>F1- Score</th><th>Support</th></tr>
<tr><td>1</td><td>Analysis</td><td>0.41</td><td>0.09</td><td>0.15</td><td>768</td></tr>
<tr><td>2</td><td>Backdoor</td><td>0.91</td><td>0.09</td><td>0.16</td><td>658</td></tr>
<tr><td>3</td><td>DoS</td><td>0.39</td><td>0.22</td><td>0.28</td><td>4909</td></tr>
<tr><td>4</td><td>Exploits</td><td>0.64</td><td>0.84</td><td>0.73</td><td>13403</td></tr>
<tr><td>5</td><td>Fuzzers</td><td>0.65</td><td>0.64</td><td>0.64</td><td>7283</td></tr>
<tr><td>6</td><td>Generic</td><td>1</td><td>0.98</td><td>0.99</td><td>17790</td></tr>
<tr><td>7</td><td>Normal</td><td>0.91</td><td>0.93</td><td>0.92</td><td>27184</td></tr>
<tr><td>8</td><td>Reconnaissance</td><td>0.91</td><td>0.75</td><td>0.82</td><td>4198</td></tr>
<tr><td>9</td><td>Shellcode</td><td>0.64</td><td>0.62</td><td>0.63</td><td>418</td></tr>
<tr><td>10</td><td>Worms</td><td>0.75</td><td>0.3</td><td>0.43</td><td>61</td></tr>
<tr><th colspan="6">Micro-Average(Precision=0.82, Recall = 0.83) and Accuracy = 0.83</th></tr>
</table>

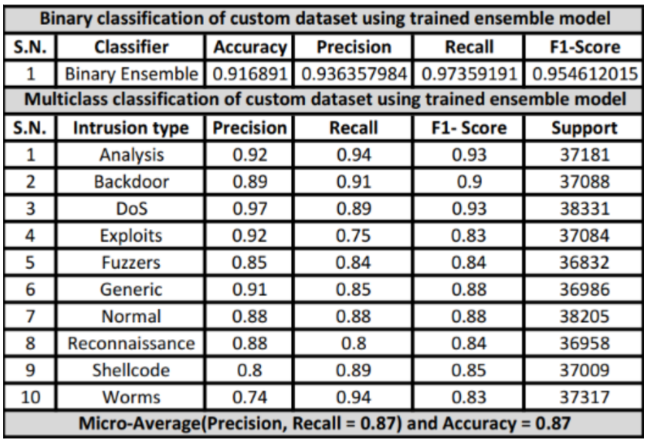
Table 3: Classification result on standard dataset (UNSW-NB15)

The data above clearly shows that the random forest and XGBoost model perform better for intrusion detection with better tracking of true positives and true negatives. Both the model show accuracy above 93 percent with precision and recall in range of 96 percent and 95 percent for random forest classifier and above 92 percent precision and above 97 percent recall for XGBoost model. The naive bayes classifier performs unsatisfactory classification with only 75 percent accuracy where as other two base models; KNN and and logistic regression show moderate classification performance with above 82 percent and 80 percent classification accuracy. The binary ensemble model shows accuracy above 91 percent with precision and recall rates above 92 percent and 97 percent respectively, and hence outperforming individual model for binary intrusion classification.

In case of multi-class intrusion classification the ensemble model shows average accuracy, precision and recall rate of 82 percent. The binary ensemble model and the multi class ensemble model are stored in local storage using *pickle* library to use the trained model for classification of custom created intrusion data set.

4.3 Classification of custom dataset using pre-trained model on standard dataset

The custom generated intrusion data set was loaded to the pre trained model on UNSW-NB15 data set to test the model's performance on custom generated intrusion data. Binary classification and multi-class classification of the custom generated data was done to observe the following results:



<table>
<tr><th colspan="6">Binary classification of custom dataset using trained ensemble model</th></tr>
<tr><th>S.N.</th><th>Classifier</th><th>Accuracy</th><th>Precision</th><th>Recall</th><th>F1-Score</th></tr>
<tr><td>1</td><td>Binary Ensemble</td><td>0.916891</td><td>0.936357984</td><td>0.97359191</td><td>0.954612015</td></tr>
<tr><th colspan="6">Multiclass classification of custom dataset using trained ensemble model</th></tr>
<tr><th>S.N.</th><th>Intrusion type</th><th>Precision</th><th>Recall</th><th>F1- Score</th><th>Support</th></tr>
<tr><td>1</td><td>Analysis</td><td>0.92</td><td>0.94</td><td>0.93</td><td>37181</td></tr>
<tr><td>2</td><td>Backdoor</td><td>0.89</td><td>0.91</td><td>0.9</td><td>37088</td></tr>
<tr><td>3</td><td>DoS</td><td>0.97</td><td>0.89</td><td>0.93</td><td>38331</td></tr>
<tr><td>4</td><td>Exploits</td><td>0.92</td><td>0.75</td><td>0.83</td><td>37084</td></tr>
<tr><td>5</td><td>Fuzzers</td><td>0.85</td><td>0.84</td><td>0.84</td><td>36832</td></tr>
<tr><td>6</td><td>Generic</td><td>0.91</td><td>0.85</td><td>0.88</td><td>36986</td></tr>
<tr><td>7</td><td>Normal</td><td>0.88</td><td>0.88</td><td>0.88</td><td>38205</td></tr>
<tr><td>8</td><td>Reconnaissance</td><td>0.88</td><td>0.8</td><td>0.84</td><td>36958</td></tr>
<tr><td>9</td><td>Shellcode</td><td>0.8</td><td>0.89</td><td>0.85</td><td>37009</td></tr>
<tr><td>10</td><td>Worms</td><td>0.74</td><td>0.94</td><td>0.83</td><td>37317</td></tr>
<tr><th colspan="6">Micro-Average(Precision, Recall = 0.87) and Accuracy = 0.87</th></tr>
</table>

Table 4: Binary and Multi-class classification result on custom created dataset using pre-trained model

In case of custom generated data set the binary model achieves classification accuracy above 91 percent with precision higher than 93 percent and recall higher than 97 percent. This shows that the pre-trained binary ensemble model not only shows higher accuracy rates on standard data set but also performs well in custom generated data set as well.

For binary intrusion classification, out of total 336028 classified data, 313645 data is classified as intrusion and 22383 data is classified normal traffic. Within that classification, 293684 instances were true positive and 14417 instances were true negative with 19961 false positive and 7966 instances of false negative.

For the multi class classification model the multi-class intrusion classification on the custom data set achieves average accuracy, precision and recall of 87 percent with higher precision, recall and f1 score in each individual intrusion classes.

5. Discussion and Analysis

First the test network was created defining the static ip routes and interfaces. The network was made capable of accepting network traffic using a virtual interface through which the intrusion and normal traffic was fed. A total of 373364 traffic instances were created in ten different traffic types. Then the voting ensemble model was built with the help of five base classifiers and the built model was trained with the the standard UNSW-NB15 dataset. The resulting binary and multi-class ensemble model was then saved in the local storage such that it could be tested in the formerly created custom dataset. The training and testing of the individual base classifier model and binary and multi-class ensemble model yield the following confusion matrices:

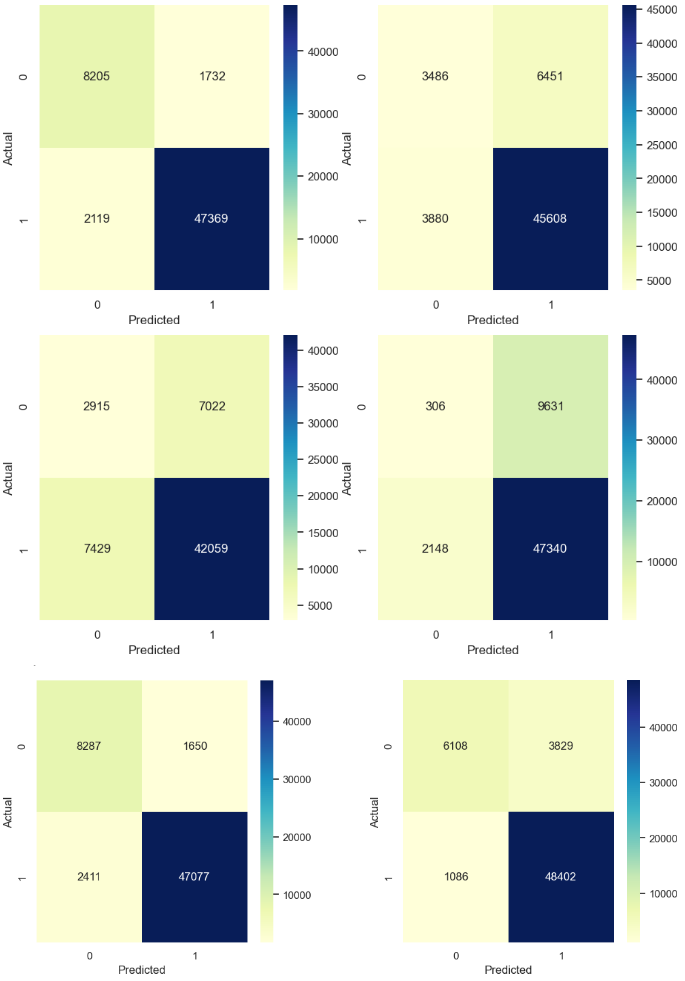


Figure 5: Confusion matrices of individual base classifiers (*top left:random forest, top right:KNN, middle left:Naive Bayes, middle right:Logistic Regression, bottom left:XGBoost* and *bottom right: ensemble model* for standard(UNSW-NB15) dataset

The first confusion matrix shows the binary classification of the standard dataset using random forest algorithm. The confusion matrix shows that 47369 counts of true positive and 8205 counts of true negative with 1732 false positive and 2119 false negatives.

Similarly the confusion matrix of KNN algorithm shows 45608 counts of true positive and 3468 counts of true neagative with 6451 counts of false positive and 3880 counts of false negatives. The KNN algorithm has not been able to track true positives with higher accuracy.

The naive bayes algorithm also does not show proper and accurate tracking of true negative, false positive and and false negative values as the confusion matrix shows high counts in false positive and false negative values of 7022 and 7429. The true positive count and true negative count with this algorithm is 42059 and 2915 respectively which shows very poor tracking to true negative values.

In case of logistic regression algorithm the classification results shows that the use of this algorithm does very poor tracking of true negative values with only 306 counts. The false positive count is also very high with 9631 count suggesting very poor performance for false positive values. The false negative value is 2148 and true positive value in this case is 47340. The use of XGBoost gives good binary classification result with proper tracking of true positive and true negative values with 47077 and 8287 counts. Further this algorithm reduces false positive and false negative values with 1650 and 2411 counts respectively thereby increasing the performance of binary intrusion classification.

The ensemble of all of those base classifiers provides further improved tracking of true positive and true negative values with reduced false positive and false negative values. For ensemble binary classification the true positive count is 48402, true negative count is 6108, false positive count is 3829 and false negative count is 1086.

In case of multi-class classification of the standard dataset the following confusion matrix was obtained:

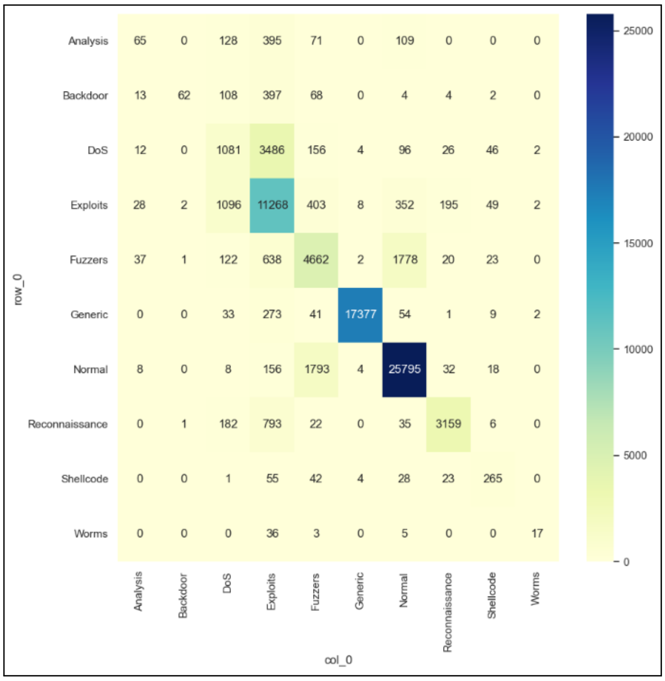


Figure 6: Multi-class classification using ensemble learning

The multi-class confusion matrix of the standard dataset (UNSW-NB15) shows good tracking of true positive values among total predicted positive in case of backdoor, generic, normal, reconnaissance and worms; intrusion classes. Further it shows good tracking of true positive among actual positive in exploits, generic, normal and reconnaissance intrusion classes. The lower values of other intrusion classes may be due to less number of such intrusion in the UNSW-NB15 dataset. The micro average of precision is 82 percent and recall is 83 percent with average accuracy of overall multi-class classification to be 83 percent.

The binary and multi-class classification of the custom generated dataset yield the following confusion matrices:

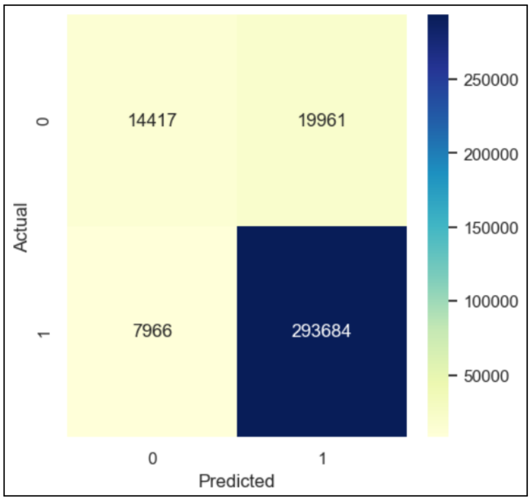


Figure 7: Binary Classification of custom generated dataset with ensemble learning

The binary classification of the custom generated dataset using ensemble learning with a pre-trained model shows good classification result with over 91 percent accuracy, 93 percent precision and 97 percent recall. The confusion matrix shows good tracking of true positives and true negatives and minimal false positive and false negative values.

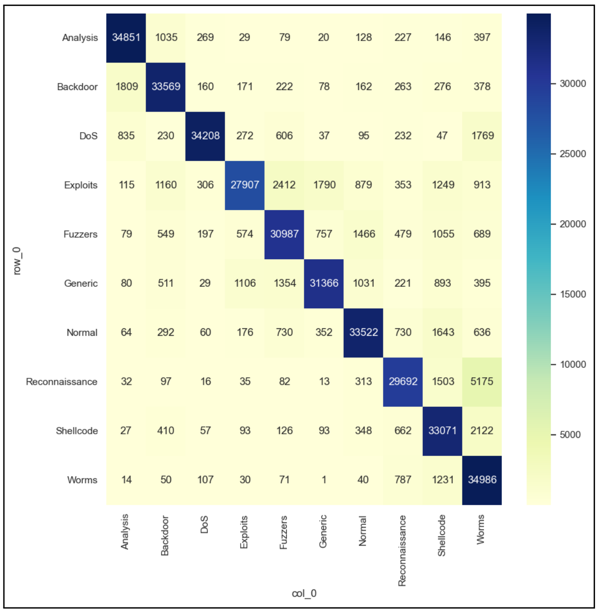


Figure 8: Multi-class Classification of custom generated dataset with ensemble learning

In case of multi-class classification of the custom generated dataset using pre-trained ensemble model, the classification results shows that the model trained on standard dataset can be used to detect intrusion on real world scenarios. The model detected ten classes of intrusion with high micro-average accuracy of over 87 percent proving its usefulness in real world scenario.

6. Conclusion and Future Work

The ensemble method can be used for various types of intrusion detection. Further different algorithms can also be tested for performance in intrusion detection scenarios. Use of different kinds of dataset can train the model for different kinds of intrusion types. The use of unsupervised learning and deep learning in domain of intrusion detection can also be explored as the future scope of intrusion detection research. As further enhancement of this proposed model, options to create a real time detection approach can be explored.

References

[1] SS Sridhar and S Prabakaran. Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing. *Cluster Computing*, 24(3):1761–1779, 2021.

[2] Albara Awajan. A novel deep learning-based intrusion detection system for iot networks. *Computers*, 12(2):34, 2023.

[3] Hooman Alavizadeh, Hootan Alavizadeh, and Julian Jang-Jaccard. Deep q-learning based reinforcement learning approach for network intrusion detection. *Computers*, 11(3):41, 2022.

[4] Yanqing Yang, Kangfeng Zheng, Bin Wu, Yixian Yang, and Xiujuan Wang. Network intrusion detection based on supervised adversarial variational auto-encoder with regularization. *IEEE access*, 8:42169–42184, 2020.

[5] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.

[6] Yingwei Yu and Naizheng Bian. An intrusion detection method using few-shot learning. *IEEE Access*, 8:49730–49740, 2020.

[7] Jie Li, Yanpeng Qu, Fei Chao, Hubert PH Shum, Edmond SL Ho, and Longzhi Yang. Machine learning algorithms for network intrusion detection. *AI in Cybersecurity*, pages 151–179, 2019.

[8] Zhen Yang, Xiaodong Liu, Tong Li, Di Wu, Jinjiang Wang, Yunwei Zhao, and Han Han. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Computers & Security*, 116:102675, 2022.

[9] S Shitharth, Pravin R Kshirsagar, Praveen Kumar Balachandran, Khaled H Alyoubi, and Alaa O Khadidos. An innovative perceptual pigeon galvanized optimization (ppgo) based likelihood naïve bayes (lnb) classification approach for network intrusion detection system. *IEEE Access*, 10:46424–46441, 2022.

[10] Ngoc Tu Pham, Ernest Foo, Suriadi Suriadi, Helen Jeffrey, and Hassan Fareed M Lahza. Improving performance of intrusion detection system using ensemble methods and feature selection. In *Proceedings of the Australasian computer science week multiconference*, pages 1–6, 2018.

[11] Naila Marir, Huiqiang Wang, Guangsheng Feng, Bingyang Li, and Meijuan Jia. Distributed abnormal behavior detection approach based on deep belief network and ensemble svm using spark. *IEEE Access*, 6:59657–59671, 2018.

[12] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, and Xinzheng He. A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5:21954–21961, 2017.

[13] Mohammed Hasan Ali, Bahaa Abbas Dawood Al Mohammed, Alyani Ismail, and Mohamad Fadli Zolkipli. A new intrusion detection system based on fast learning network and particle swarm optimization. *IEEE Access*, 6:20255–20261, 2018.

[14] Yang Jia, Meng Wang, and Yagang Wang. Network intrusion detection algorithm based on deep neural network. *IET Information Security*, 13(1):48–53, 2019.

[15] Kaiyuan Jiang, Wenya Wang, Aili Wang, and Haibin Wu. Network intrusion detection combined hybrid sampling with deep hierarchical network. *IEEE access*, 8:32464–32476, 2020.

[16] Gozde Karatas, Onder Demir, and Ozgur Koray Sahingoz. Increasing the performance of machine learning-based idss on an imbalanced and up-to-date dataset. *IEEE access*, 8:32150–32162, 2020.

[17] Ritesh K Malaiya, Donghwoon Kwon, Sang C Suh, Hyunjoo Kim, Ikkyun Kim, and Jinoh Kim. An empirical evaluation of deep learning for network anomaly detection. *IEEE Access*, 7:140806–140817, 2019.

[18] Xueqin Zhang, Jiahao Chen, Yue Zhou, Liangxiu Han, and Jiajun Lin. A multiple-layer representation learning model for network-based attack detection. *IEEE Access*, 7:91992–92008, 2019.

[19] Peng Wei, Yufeng Li, Zhen Zhang, Tao Hu, Ziyong Li, and Diyang Liu. An optimization method for intrusion detection classification model based on deep belief network. *Ieee Access*, 7:87593–87605, 2019.

[20] Xianwei Gao, Chun Shan, Changzhen Hu, Zequn Niu, and Zhen Liu. An adaptive ensemble machine learning model for intrusion detection. *Ieee Access*, 7:82512–82521, 2019.

[21] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers and Electrical Engineering*, 102:108156, 2022.

[22] Haipeng Yao, Danyang Fu, Peiying Zhang, Maozhen Li, and Yunjie Liu. Msml: A novel multilevel semi-supervised machine learning framework for intrusion detection system. *IEEE Internet of Things Journal*, 6(2):1949–1959, 2018.

[23] Farrukh Aslam Khan, Abdu Gumaei, Abdelouahid Derhab, and Amir Hussain. A novel two-stage deep learning model for efficient network intrusion detection. *IEEE Access*, 7:30373–30385, 2019.

[24] Dimitrios Papamartzivanos, Félix Gómez Mármol, and Georgios Kambourakis. Introducing deep learning self-adaptive misuse network intrusion detection systems. *IEEE access*, 7:13546–13560, 2019.

[25] I Ahmad, Q. E. U. Haq, M. Imran, M. O. Alassafi, and R. A. Alghamdi. An efficient network intrusion detection and classification system. *Math*, 10:530–534, 2022.

[26] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.