The Use of Machine Learning in The Detection of Anomaly Network Intrusion

Akshat Behera
A20516439
abehera2@hawk.iit.edu

Nagarjuna Bolla
A20524548
nbolla@hawk.iit.edu

Intermediate Project Report

1 Project Introduction

In the modern age of digitization, concerns regarding the safety and security of personal and professional data have reached an all-time high. Network Intrusion Detection has emerged as a critical component of cybersecurity, with the primary goal of identifying and preventing network vulnerabilities from being exploited by malicious actors. However, as attackers continue to refine their methods to bypass detection, traditional rule-based approaches have proven to be ineffective in identifying sophisticated and novel attacks.

To combat this issue, Machine Learning has become a crucial tool in Network Intrusion Detection. By utilizing large datasets and identifying patterns in network traffic, Machine Learning approaches can play a vital role in detecting potential threats. This project aims to develop an effective Machine Learning-based approach for Network Intrusion Detection using the UNSW-NB15 dataset. The dataset contains both normal and attack traffic, and the aim is to accurately classify traffic into different attack categories.

The proposed approach seeks to build on existing research by exploring the potential of Neural Networks (NNs), such as Neural Network (NN) using MLP Classifier, Convolutional Neural Networks (CNNs), and Recurrent Convolutional Neural Networks (RCNNs), for Network Intrusion Detection. The project will also investigate the benefits of feature selection methods in improving classifier performance. Finally, the project aims to compare the effectiveness of NNs with traditional classifiers.

Preprocessing of the dataset is the first step, which involves Extraction and Cleaning of the Data. The UNSW-NB15 dataset contains missing values and categorical variables, which need to be addressed before feeding the data into Machine Learning models. Exploratory Data Analysis (EDA) is also essential to understand the distribution of the features, correlation among the features, and to identify any outliers in the data.

The act of feature selection is to determine the most pertinent features that will advance the accuracy of the classifiers. The scheme is to employ a score based on Mutual Information to determine the top k features. Varied types of Neural Networks will be chosen and juxtaposed with established classifiers to determine their efficacy. The classifiers' effectiveness will be determined through various metrics such as Accuracy, Precision, Recall, F1-Score, and AUC-ROC.

In summary, the primary objective of this project is to experiment with advanced methodologies in Network Intrusion Detection, thereby enhancing the classifier accuracy. A potent Network Intrusion Detection system, proficient at identifying even the most innovative and complex attacks, could be the ultimate product of this research, and could bring untold benefits to organizations worldwide.

2 Problem Description

Cybersecurity is one of the major concerns for organizations worldwide. Network Intrusion Detection is a crucial component of cybersecurity, which detects and mitigates the attempts of attackers to exploit vulnerabilities in a network. The detection of these attacks is a challenging task as the attackers are continuously modifying their techniques to bypass the detection methods.
Traditional rule-based approaches for detecting network intrusions are not effective as they have limitations in identifying novel and sophisticated attacks. Machine Learning approaches can play a vital role in detecting these attacks as they can learn from large datasets and identify patterns in network traffic.
The proposed project aims to develop an effective Machine Learning-based approach for Network Intrusion Detection using the UNSW-NB15 dataset. The dataset contains both normal and attack traffic, and the goal is to classify the traffic into different attack categories accurately.

3 Description of the Dataset used in the Project

The dataset used in the project is called UNSW-NB15 dataset (also available at Kaggle), which is a network traffic dataset that contains labeled traffic data generated by a network simulator. The data contains a total of 49 features, which include nominal, integer, float, timestamp, and binary types. The Dataset has been uploaded and is available at the our Project Github Repository.

With a total of 49 features extracted from each network connection but our primary datasets which we use, only have 45 features in use. The first features are not included in those datasets, the dataset contains both benign and malicious traffic. Protocol type, source and destination IP addresses, source and destination port numbers, connection duration, number of bytes sent and received, and many more are among the features.

A training set and a testing set were created from the UNSW-NB15 dataset. There are 175,341 records in the training set and 82,332 records in the testing set. The training and testing sets are provided in separate CSV files alongside the dataset. Depending on the type of traffic, the files are further split into several file categories, such as Normal, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, etc.

```
df = pd.read_csv('UNSW_NB15_training-set.csv')
```

```
df.shape
```

```
(175341, 45)
```

```
df = pd.read_csv('UNSW_NB15_testing-set.csv')
```

```
df.shape
```

```
(82332, 45)
```

Figure 1: Training and Testing Dataset CSV File - Implemented on Seperate .ipynb Notebooks

Overall, the UNSW-NB15 dataset is a valuable resource for network intrusion detection researchers and practitioners. Because of its real-world nature and diverse range of network traffic, it is an excellent benchmark dataset for testing and comparing the performance of various machine learning algorithms and intrusion detection systems.

The first four features, namely srcip, sport, dstip, and dsport, represent the source IP address, source port number, destination IP address, and destination port number of the network traffic, respectively it's not included in our primary datasets (training and testing).The features begins with the id and The fifth feature, proto, represents the transaction protocol used in the network traffic, and the sixth feature, state, indicates the state and its dependent protocol.

The next eight features are numerical values that measure various aspects of the network traffic. The dur feature represents the total duration of the traffic, while sbytes and dbytes represent the number of bytes transmitted from the source to the destination and from the destination to the source, respectively. The sttl and dttl features represent the source to destination and destination to source time to live values. The sloss and dloss features represent the number of packets retransmitted or dropped by the source and destination, respectively. Finally, Sload and Dload represent the source bits per second and destination bits per second, respectively.

The number of packets transported from the source to the destination and from the destination to the source is shown by the next four attributes, which are integers. The source and destination TCP window advertisement values are represented by the following two characteristics, swin and dwin. The source and destination TCP base sequence numbers are represented by the following two features, stcpb and dtcpb. Smeansz and dmeansz, the following two attributes, stand for the average flow packet size transmitted by the source and destination, respectively.

The next two features, trans_depth and res_bdy_len, stand for the pipelined depth of the http request/response transaction inside the connection and the actual uncompressed content size of the data transported from the server's http service, respectively.

The next two features, Sjit and Djit, represent the source and destination jitter, respectively, and are measured in milliseconds. The next two features, Stime and Ltime, represent the record start time and record last time, respectively, and are measured in timestamps. The next two features, Sintpkt and Dintpkt, represent the source interpacket arrival time and destination interpacket arrival time, respectively, and are measured in milliseconds.

The next three features, tcprtt, synack, and ackdat, are float values that measure different aspects of the TCP connection setup time. tcprtt represents the TCP connection setup round-trip time, synack represents the time between the SYN and the SYN_ACK packets, and ackdat represents the time between the SYN_ACK and the ACK packets.

The next three features, is_sm_ips_ports, ct_state_ttl, and ct_flw_http_mthd, are binary and integer values that measure different aspects of the network traffic. is_sm_ips_ports is binary and takes a value of 1 if the source and destination IP addresses and port numbers are equal, and 0 otherwise. ct_state_ttl is an integer that represents the number for each state (6) according to specific ranges of values for source/destination time to live (10) and (11). ct_flw_http_mthd is an integer that represents the number of flows that have methods such as Get and Post in the http service.

The next three features, is_ftp_login and ct_ftp_cmd, are binary and integer values that measure different aspects of the FTP service. is_ftp_login is binary and takes a value of 1 if the FTP session is accessed by user and password and 0 otherwise. ct_ftp_cmd, represents the number of flows that have an FTP command, respectively.

The forty-first, forty-second, forty-third, and forty-fourth features, ct_srv_src, ct_srv_dst, ct_dst_ltm, and ct_src_ltm, represent the number of connections that contain the same service and source/destination address in 100 connections according to the last time.

The "ct_src_ltm" attribute represents the number of connections from the same source address in 100 connections, indicating the frequency of traffic coming from a specific source.

The "ct_src_dport_ltm" attribute represents the number of connections from the same source address and destination port in 100 connections, indicating the frequency of traffic coming from a specific source to a specific destination port.

The ct_dst_sport_ltm attribute represents the number of connections to the same destination address and source port in 100 connections, indicating the frequency of traffic going to a specific destination from a specific source port.

The dataset includes a categorical variable in column 48 indicating the type of attack (if any) associated with each connection, with nine categories such as Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. Finally, column 49 is a binary variable indicating whether the connection was classified as a normal transaction or an attack.

4 Work Completed So Far

As we know that from UNSW Dataset, a training set and a testing set were created from the UNSW-NB15 dataset. There are 175,341 records in the training set and 82,332 records in the testing set. The training and testing sets are provided in separate CSV files alongside the dataset. We use each of the datasets in seperate notebooks i.e., We have divided the code for Training dataset in one notebook and for the Testing dataset in another notebook.

Below are Milestones, We have Completed:

1. Preprocessing of the Dataset: Here, We begin with Extraction and Cleaning of the Data. The UNSW-NB15 dataset (both in Training and Testing Datasets) contains missing values and categorical variables, which needed to be addressed before feeding the data into the Machine Learning models. Here We, preprocessed the data by dropping unwanted columns, encoding categorical variables by using LabelEncoder(), and standardizing numerical variables by using StandardScaler(). This is done in both the notebooks of the Training Dataset and Testing Dataset.

2. Exploratory Data Analysis (EDA): EDA is essential to understand the distribution of the features, correlation among the features, and to identify any outliers in the data. In accordance to our plan we have performed EDA using visualizations such as histograms, heatmaps, and boxplots. This is done in both the notebooks of the Training Dataset and Testing Dataset. These visualizations obtained gives us a more clearer picture on understanding the dataset and it's distribution across various factors.

3. Selection of the Features: Feature selection aims to select the most relevant features that can improve the performance of the classifiers. Here, In this project in accordance to our plan we have used the Mutual Information Score to select the top k features (k=10) using SelectKBest. We have utilized a methodology known as Mutual Information Score, which is a dependable and streamlined approach for recognizing pertinent features. Through this approach, we can enhance the efficiency of our Machine Learning models by decreasing the size of the input data and eliminating noise and redundancy. We have implemented this approach in both the Training and Testing Dataset notebooks.

4. Selection and Training of the Models: In accordance to our plan was to select different types of Neural Networks, such as CNNs, RCNNs, and NNs (MLP), and compare their performance with traditional classifiers such as Random Forest, Decision Tree, XGBoost, SVM and Logistic Regression.
   So far, we have completed the Selection and Trained the Traditional Classifiers/Supervised Learning models for Logistic Regression, Decision Tree, Random Forest, SVM and XGBoost. We have obtained the performance results for these models. This is done in both the notebooks of the Training Dataset and Testing Dataset.

5. Evaluation of the Models (Supervised): The performance of the Traditional classifiers/Supervised Learning Models has been evaluated using metrics such as Accuracy, Precision, Recall, F1-Score, and Precision Recall and ROC Curves Graphs and AUC Score Plots. We have used the classification report and confusion matrix for each of the supervised models/traditional classifiers to identify the misclassified samples and analyze the results. This is done in both the notebooks of the Training Dataset and Testing Dataset.

```
Logistic Regression Accuracy: 0.8963747314791932
Decision Tree Accuracy: 1.0
Random Forest Accuracy: 1.0
SVM Accuracy: 0.9997148451609223
XGBoost Accuracy: 1.0
```

```
Logistic Regression Accuracy: 0.931255060728745
Decision Tree Accuracy: 1.0
Random Forest Accuracy: 1.0
SVM Accuracy: 0.9990283400809716
XGBoost Accuracy: 1.0
```

Figure 2: Accuracy of Supervised Models - Training V/S Testing Dataset

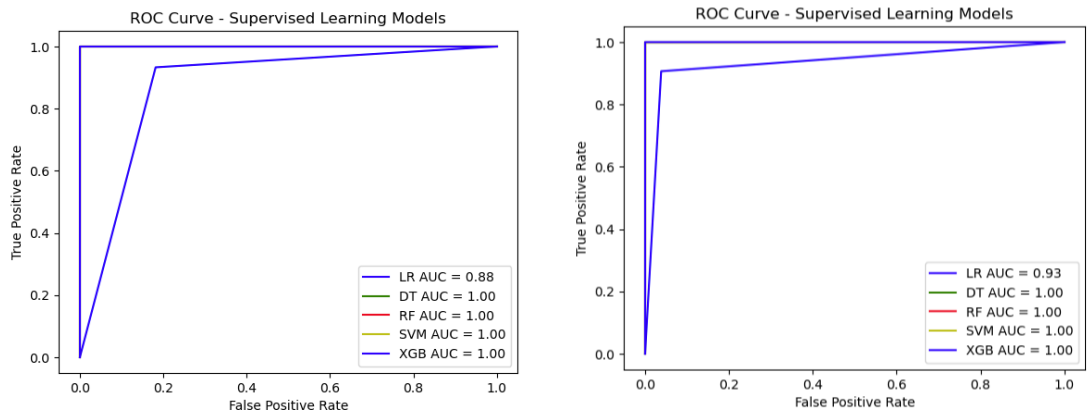

Figure 3: ROC Curves of Supervised Models - Training V/S Testing Dataset

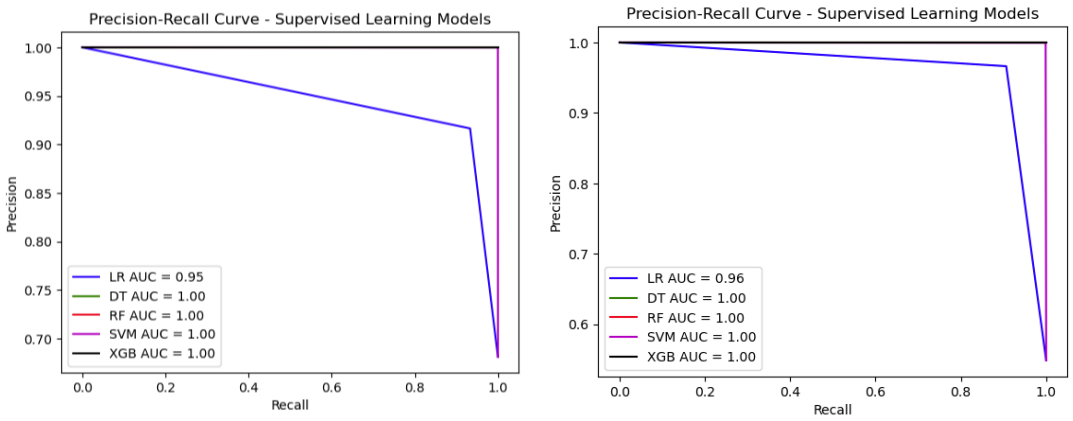

Figure 4: Precision Call Curves of Supervised Models - Training V/S Testing Dataset

5 What Remains To Be Done

Now, the work that remains to be done is as Follows:

1. Selection and Training of the Models (Unsupervised): After the feature selection , we move on to the Model Selection and Training for the Unsupervised Learning models. As we know that, In accordance to our plan was to select different types of Neural Networks, such as CNNs, RCNNs, and NNs (MLP), and compare their performance with traditional classifiers such as Random Forest, Decision Tree, XGBoost, SVM and Logistic Regression which we have already obtained.

   So, In the Unsupervised Models/Classifiers, We Select the following models such as: NN(MLP Classifier), CNN (Convolutional Neural Networks) and RCNNs. After Selecting these models, we then train the model in both the dataset notebook files i.e., Training and Testing dataset, using the particular dataset and the features obtained from the feature selection.

2. Evaluation of the Models (Unsupervised): After that, The performance of the Unsupervised Learning models shall be evaluated using the following metrics such as: Accuracy, Precision, Recall, F1-Score, and Precision Recall and ROC Curves graphs and AUC score plots for each of the models.
   We can use the classification report to provide us with a summarized information on the models and Confusion Matrix for each of the unsupervised model to identify the misclassified samples and analyze the results. This is to be done in both the notebooks of the Training Dataset and Testing Dataset.
   Finally, After we evaluate the performance of the Unsupervised models. We then can compare it's performance with the already evaluated Supervised models performances and draw the conclusion on the project, with the comparison analysis.