Evaluation of Machine Learning and Deep Learning Algorithms on Network Intrusion Detection

MSc Research Project
Cybersecurity

Oluwaseun Olaniyan
Student ID: X19231423

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	…….OLANIYAN OLUWASEUN ABIOLA……		
Student ID:	……X19231423……		
Programme:	……Cybersecurity……	**Year:**	……2020/2021……..
Module:	……MSC……		
Supervisor:	……Michael Pantridge……		
Submission Due Date:	…16/08/2021……		
Project Title:	……		
Word Count:	…… **Page Count**……		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	……OLANIYAN OLUWASEUN ABIOLA……
Date:	……15/08/2021……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

ABSTRACT

Due to the introduction of the devices for networking with the fast internet development in earlier years, the safety of the networks has developed to be important in this contemporary age. (NIDS) Network Intrusion Detection Systems are used in identifying unapproved, unacquainted and traffic that is suspicious through networks. This project pursues the combination of the two commonly known network intrusion detection types that are, misuse detection and anomaly detection through the design of a hybrid model that classifies a network traffic first either as benign or intrusive. When the traffic is established as intrusive, the model additionally detects the intrusive traffic category travelling throughout the network. Furthermore, the research proposes deep learning and machine learning algorithm usage in determining the quickest and utmost precise algorithm for network intrusions detection.

Keywords—machine learning, Anomaly detection, NIDS, deep learning, misuse detection

I. INTRODUCTION

Network intrusion is defined as an act performed to violate the Availability, Integrity and Confidentiality of the info transferred through the network. It's additionally defined as an action done to avoid a device's or network's safety methods. The action of evaluating and observing the network actions for such infringements is termed as intrusion detections [1]. The system for (IDS) Intrusion detection is whichever setup used in detecting unauthorised or new entry into the system. Nevertheless, IDS may be only used in identifying the abnormalities as they are being performed or when execution have been performed. Networking administrators first implements the intrusion detections thru the monitoring of users network data usage thru a terminal [2]. Instances of such disturbances were characterized through a person who was expected to be on holiday logged into the local network or a regularly used network printer showing high usage. Though it was demonstrating being efficient then, this earlier intrusion detection method was habitually tiresome and unscalable recurring. The increasing usage of the internet throughout the past years contributed more to the increasing cyberthreats numbers throughout the world. In 2011, CISCO stated that there were above ten billion linked devices over the internet and the number was seen to rise by nearly 5% by the subsequent decade [3]. This places into view the extent of devices exposed to attacks by invaders. These invaders may be in 2 methods, as acknowledged by [4], namely outer intruders and in-house.

- In-house intruders imply persons having lawful authorizations of connecting a network but rudely abuses the privileges, typically for their greedy gains. This is normally tougher to find since the entities have authorised right already to join the network and get entry to specific data.

These invaders are mainly recognized both when releasing sensitive data or when accessing certain data, which is separate from their authority.

- Outer intruders are people who do not have authorized approval being on a certain network, but enter the network, frequently through the distribution of phishing mails to folks inside the network, thus exploits weaknesses in the network or through password protected systems hacking by means of brute force.

This study is organised into the following sections: The Literature Review discusses in-depth mechanisms of an intrusion detection system, the of intrusion detection methods of systems, and functions, strengths and drawbacks of deep learning and machine learning algorithms within network intrusion detection, the research question which is how can machine learning and deep learning algorithms be used in intrusion detection systems? Other subsections are the research Methods which details the methodology used to complete the project, the design specification, evaluation, discussion, and conclusion.

The purpose of this analysis is answering the research's problem: How can deep learning and machine learning algorithms be used to identify network's intrusion attacks?

The following objectives will also be explored:

- Using machine learning algorithms as a network intrusion detection system
- Create a network IDS using deep learning algorithms
- To compare the results of the deep learning algorithms with the machine learning algorithm depending on the accuracy scores recorded for each.

II. LITERATURE REVIEW

Safety of the network is topmost on the list of most company's safety methods. It comprises the procedures used in securing network's set-ups for physical and cyber-attacks. The farther the businesses and organisations are moving their services online, the more they are visible to superior threats of attacks executed by hackers. These attacks frequently target paralyzing the business's functioning infrastructures, holding properties to ransom or absolute extortion. Consequently, businesses are at threat of possibly missing their revenues due to the attacks [5]. Some instances of the extortions are stated below [6]:

1. (DoS) Denial of Service

Form of an attack in which the attackers make a networking device or services unavailable to its anticipated users that is normally done thru network flooding using extreme and pointless requests [7].

2. Man-in-the-Middle

Kind of an attack in which the attackers secretly communicate then, in certain situations, adjusts the data moving amongst 2 personalities or devices who may believe they are directly interacting with each other. This attack method may be particularly dangerous if the involved people are sharing secret data [6].

3. Malware

The kind of attacks, which comprises installation of (malware) on an intended machine to access and recover personal/valued info kept on the device and in some instances, damaging the info or the whole devices. Malware is constantly transferred to commit damaging intent, regularly for some financial compensation [6].

4. Port Scan Attack

The category of attacks is aimed at getting data of the category of a device within the network. It works thru various ports scanning in a network then returns the conditions of the ports. On discovering an open port, invaders may then obtain data like the running port services, operators who opened the services, services that need or do not require confirmation, etc.

5. Botnet

The form of attack caused by the malware, which makes network devices to be manipulated by the invader exclusive of the awareness of the device user. The invaders thus take control of these robots (Bots) and make it corrupt or interrupt services on other devices on a network.

A. Intrusion Detection System

Used in detecting safety breaks in form of invasions in a network. IDSs works by observing the network activities in an attempt to spot glitches in network activity. It is performed using the theory, in which the characters that separate traffic that is malicious varies to the genuine traffic. IDSs began from the mid 1980s [8]. Networks executives originally instigated intrusion detection through scrutinizing the clients' network data usage thru a terminal [2]. Although this method was successful at that period, it was additionally established to be tiresome, repetitive and unscalable.

There are distinctive sets used in IDSes; however [9] detailed 2 classifications of IDS and [10] described 3 classifications of IDSes, the tow findings, as with other findings similar to [11] and [12] shared 2 classifications of IDS, which are misuse detection (signature/ knowledge-based detection) and anomaly detection.

i. Anomaly Detection

Usually identified as a shift from standard or normal behaviour. It applies to network traffic; Anomaly detection commonly works by detaining the usual traffic and operation benchmark, which takes place on the network. They then compute the existing network traffic situation alongside this benchmark so as it identifies characters that are not in the normal traffic. Such methods will execute

fine while trying to identify new risks, which have been deliberately designed to avoid discovery by IDSes [13].

ii. MISUSE DETECTION

The signature-based or misuse detection use the signatures or conduct of formerly renowned vulnerabilities or attacks in the system in determining invasions. It is additionally denoted as knowledge-based detection since the system holds the information of some of previously detected threats and associates the signature of incoming traffic with the existing one in the knowledge base [14].

Although authors maintained that the massive relationships in the 2 detection modes, anomaly-based detection technique is regularly extra correct in discovering latest threats that can be overlooked thru the misuse-based detection technique.

Subsequently, the proposed research purposes to creating a model, which joins both detection techniques through identification of a reference of normal traffic then collating a record of identified attacks. This model will be trained to discriminate amongst intrusive traffic and normal traffic, whereas having the ability to detect the intrusion types detected within the network.

B. CLASSIFICATION ALGORITHMS FOR INTRUSION DETECTION

i. Decision tree

Decision Tree (DT) is one of the most essential supervised machine learning algorithms used in the classification and regression of specific data sets by using a series of rules. It is an algorithm that groups data using a chain of rules. As the name suggests, the model has a tree-like structure making it is easy to interpret. The DR algorithm is able to omit redundant and unrelated features automatically. The learning process in DT comprises tree pruning, tree generation and feature selection [1]. The algorithm can select the most apt features separately and creates child nodes from the root node during its training [30]. The decision tree is an essential classifier. Extreme gradient boosting (XGBoost) and random forest are some of the superior algorithms that comprise manifold decision trees.

ii. K-Nearest neighbour (KNN)

The KNN algorithm works based on multiple hypothesis. In case sample's neighbors are of the same class, the sample is probable to belong to the class. For that reason, the result of classification is merely linked with top-k nearest neighbors. The performance of KNN model is vastly affected by the parameter k [29]. The model becomes more complex when k is smaller thus the greater the risk of over fitting. In contrast, the larger the k, the simpler the model and the weaker the fitting capability.

iii. Support vector machine (SVM)

Support vector machine (SVM) is a monitored machine learning algorithm that uses the notion of a hyperplane with the largest separation field in a multi-dimensional feature space. SVMs are designed to get a maximum margin separation hyper plane found in the n-dimension feature space [29]. The

results that SMVs achieve is satisfying even in a small-scale training session because a small number of support vectors determine separation hyperplane. SVMs, nevertheless are so responsive to noise near the hyperplane [31]. Additionally, SVMs are capable of resolving linear problems thus making them reliable.

Artificial Neural Network (ANN)

Similar to DT and KNN, ANN is a monitored machine learning algorithm, which is based on the functions of the human brain nervous system. ANN is designed in a way that it is able to imitate how the human brain works. It has layers such as output layer, many hidden layers and input layer. All the units in the nearby layers are completely connected. A single ANN comprises a massive number of units which can hypothetically estimate random functions therefore, it has robust fitting capability, specifically for functions which are nonlinear [29] Training ANN consumes a lot of time due to its intricate model structure. As a result, back propagation algorithm is used to train ANN

Logistic regression

Logistic regression (LR) is a robust and complete supervised classification technique. This is a linear model algorithm. It is an algorithm that calculates the possibility of various classes through a parametric logistic distribution as computed in the formula below [32].

$$P(Y = kjx) = \frac{e^{w]k\ x}}{1 + \mathring{a}_k^{K\ 1}\ e^{w]k\ x}}$$

Here, k =1, 2… k-1. *X* as a sample is grouped in maximum probability class. It is very simple to construct LR model, and model training is effective.

Random forest

Random Forest (RF) is an ensemble classifier composed of a set of DTs, just like a forest is a collection of trees. DTs that are too deep usually contribute to over-fitting of the training data, which results in a large dispersion of the classification results, and small changes in the input data [33]. The DTs appear quite sensitive to their training data, making them prone to errors in the test data set. Distinct DTs of a Random Forest are trained using the data set components. The input vector of the sample must be transmitted with each DT in the forest to classify a novel sample [32]. Resultantly, each DT considers

varied parts of this input vector and outputs a ranking result. Figure 1 shows the Random Forest algorithm.

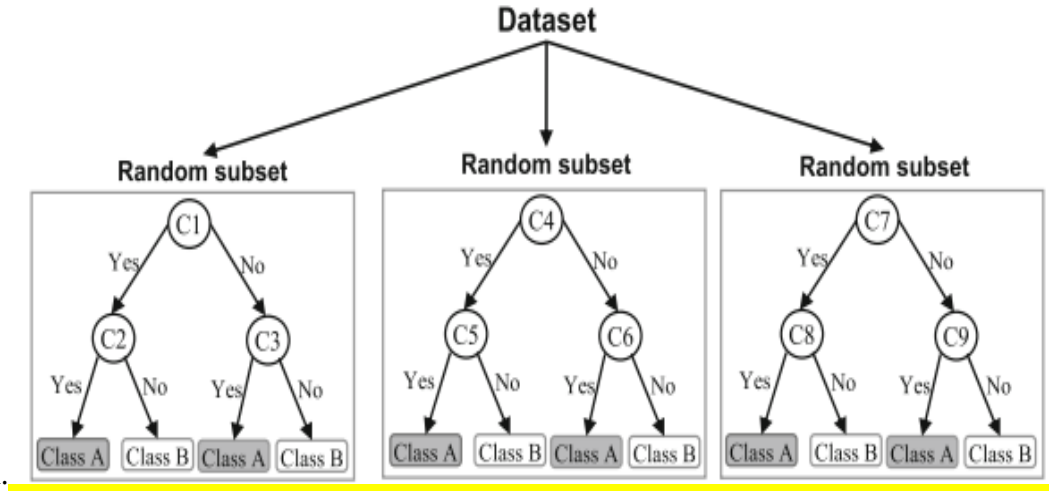


Fig. 1: Random Forest Algorithm Illustration

iv. Naïve Bayes (NB)

Naive Bayes (NB) refers to a categorization method grounded on Bayes' paradigm. The theorem defines the likelihood of an event on the basis of previous knowledge of the conditions associated with the event [32]. The classifier presumes that a certain characteristic of one class is not directly linked to another feature. Figure 2 illustrates how the NB technique operates.

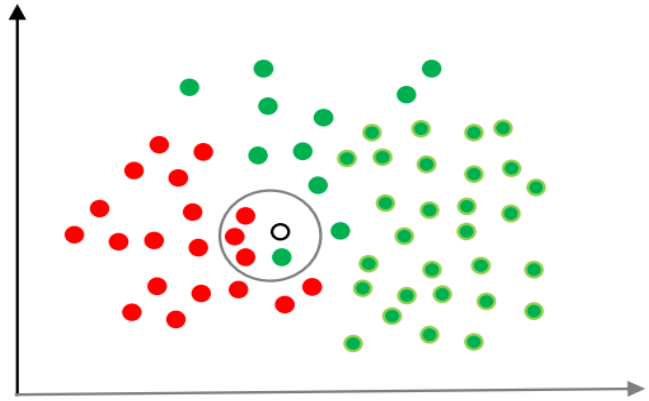


Fig. 2: Depiction of Operation of NB Method. The 'white' circle represents the novel sample instance that requires classification as either to 'red' or 'green' category.

a) Unsupervised machine learning algorithm

i. *K-mean clustering*

The clustering is the process of dividing data into meaningful groups by placing highly similar information within the same category. K-mean clustering is a major machine learning algorithms based on iterative centroids, which can learn unsupervised. *K i*ndicates the number of focal points in the data set [29]. To assign certain data points to a group, a distance is usually computed.

C. DEEP LEARNING ALGORITHMS

D. SUPERVISED DEEP LEARNING ALGORITHMS

RECURRENT NEURAL NETWORKS (RNN)

RNN expands the capability of traditional feed-forward neural networks to model sequence data. The approach is composed of input, hidden and output blocks. Hidden blocks are regarded as storage elements [29]. For decision-making, each RNN block is based on its current input and the output of the preceding input.

DEEP NEURAL NETWORK (DNN)

DNN is a fundamental deep learning structure, which can be utilized to train models on multiple levels. The method consists of input and output layer, as well as diverse hidden layers [34]. DNN is applied in modelling complex nonlinear functions. More hidden layers increase the abstraction degree of the model and improve its capacity.

CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional neural network (CNN) is a deep learning structure that is most appropriate for data stored in arrays. CNN is composed of an input layer, a set of convolutions and grouping surfaces for feature extraction, and a fully linked layer and a softmax classifier in the classification level [29]. The method is widely used in computer vision. In terms of IDS, CNN is utilized to extract monitored features and for classification.

FEED FORWARD NEURAL NETWORK (FFNN)

FFNN is applied in modelling macroscopic material behaviour utilizing strains (input) and stress (output). A key merit of the model is that the training data needed by the neural network can be obtained directly from the experimental data [35]. In FFNN, the quantity inputs and outputs is specified, and the knowledge of historical variables is ignored. Based on the complexity of the training data, the FFNN architecture should be determined.

E. PAST STUDIES

Diverse research directions on intrusion detection systems using deep learning and machine learning techniques exist. [38] proposed an IDS based on RNN in the context of binary and multi-class categorization of the NSLKDD dataset. The model was experimented with distinct numbers of concealed nodes and learning frequencies. The findings showed that varied learning rates and the number of hidden nodes affected the model's accuracy. For binary and multi-class scenarios, 80 hidden nodes and learning rates of 0.1 and 0.5 were used to achieve the highest accuracy. The main disadvantage of this study was the rise in the number of calculations, which led to a longer training time of the model. The article also lacked a comparison of the performance of the suggested model with other different deep learning approaches.

[39] recommended an IDS based on deep AutoEncoder and ML methods. Only the AutoEncoder coding part was used to make the model computational and time efficient, while working asymmetrically. Two non-asymmetric deep AEs with three hidden layers were superimposed on each other. Experiments were conducted on multi-class categorization scenarios using the KDD Cup'99 and

NSL-KDD datasets. However, due to the lack of data for training the model, it was ineffective in detecting R2L and U2R attacks.

[40] advanced an independent misuse detection system that combine the benefits of self-learning and the MAPE-K models. The researchers utilized sparse AutoEncoder for unsupervised learning algorithms to gain insight into practical functions when performing planning operations in MAPE-K. The experiment was conducted using data sets KDD Cup'99 and NSL-KDD. The primary disadvantage is the lack of accuracy in detecting attack categories for U2R and R2L categories.

This study purposes to add value to previous research in the field of information systems security by proposing the most efficient algorithms that can be used to detect intrusion in network systems.

III. METHODOLOGY

The study was conducted using the Knowledge Discovery in Databases (KDD) approach. This approach features procedures such as data selection/ transformation/, evaluation, pre-procession collection and modelling. Some of these processes are explained below:

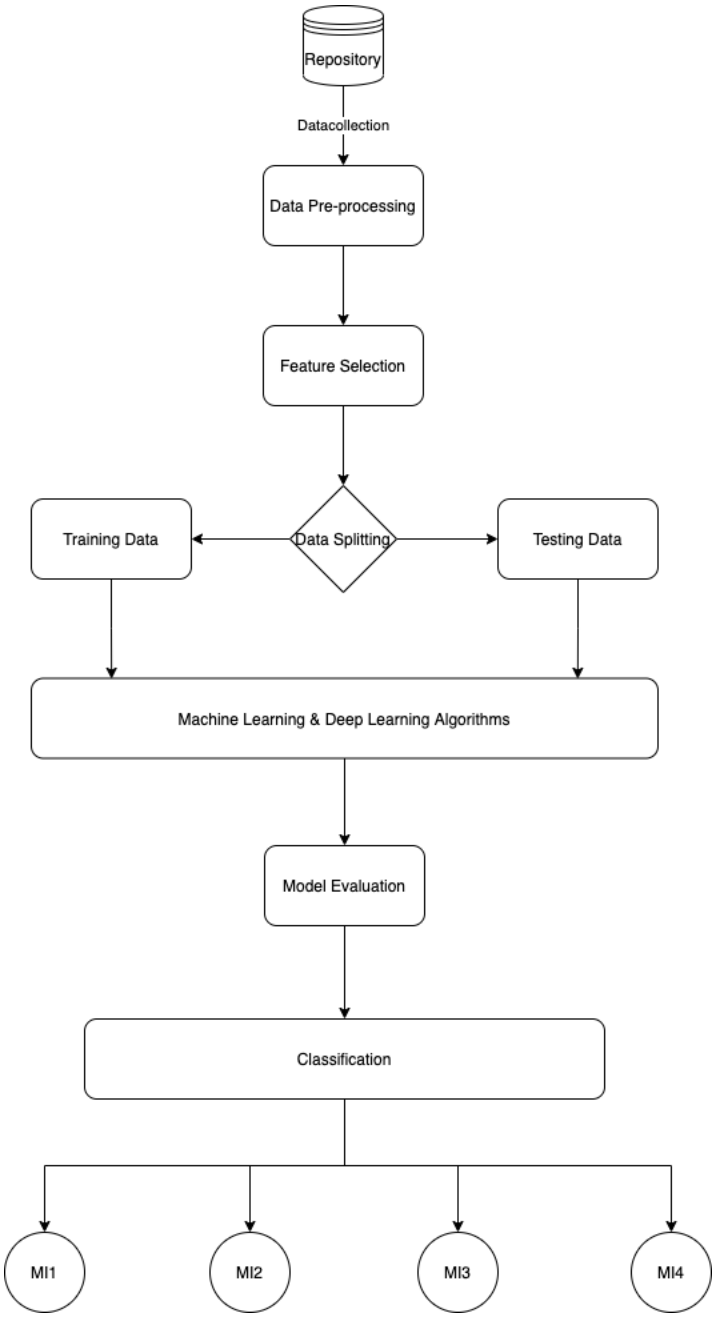


Figure 1: Proposed Methodology for Intrusion Detection System

DATA COLLECTION

The study used a dataset that was already available in a public data repository. This was to avoid ethical issues that could have arose during the execution stage. The data was downloaded and locally stored to be used later for processing. Personal information was excluded from the dataset to ensure that participants remain anonymous

DATA PRE-PROCESSING

The downloaded dataset was analysed to determine the specific distributions. This process is necessary as it helps identify the available classes with the respective network intrusion classes. The goal of performing data pre-processing is to ensure balancing of the dataset in case there is any unbalanced distribution which is a common classification problem [24]. To handle such issues, the research would use either under-sampling or oversampling techniques. While under-sampling involves the random selection of fewer observations from the majority class to match with those in the minority, oversampling is a technique that involves record duplication from the minority class to get a match with the majority class [25].

The two techniques despite being efficient in ensuring data balancing have some few challenges. For instance, there is the risk of data loss in under-sampling which can affect the overall accuracy of the experiment. On the other hand, oversampling can lead to duplication of data which also causes model overfitting [25].

The study also took extra steps in pre-processing to ensure the dataset was well prepared for the experiment. These steps included the null values in the dataset which could affect the performance of the algorithms. Null values are also known to reduce the model's efficiency, make data analysis difficult, and can lead to bias due to the missing information [26]. To handle null values, the study opted to take a number of strategies such as deleting the affected values, replacing, or using the closest fit method [27].

FEATURE SELECTION

Feature selection is an approach to data mining that involves choosing relevant features from a large number of dimensions in a dataset [28]. This study used the Lasso and tree-based feature selection methods from the "SelectFromModel" library in python. The feature works by reducing the number of features in a data set through the elimination method. In other words, features that are rendered useless and do not impact the model's performance are removed while the ones that affect performance remain.

CLASSIFICATION ALGORITHMS

The study used two machines learning and one deep learning algorithm. The selection of machine learning algorithms include:

- Random Forest
- Support Vector Machine (SVM)

The deep learning algorithm used was:

- Artificial Neural Network (ANN)

The goal is to optimise the performance of each algorithm and evaluate the results based on the accuracy scores.

MODEL EVALUATION

The study used different evaluation metrics namely confusion plot, area under curve, accuracy, precision and recall to train the model after the testing process was complete.

ROC Area Under Curve

This metric determines the true positive in relation to the false positive rate. The graph plots a baseline performance. Models with incorrect predictions are represented with AUC value between 0.00 and 1.00 The AUC value is 0.00 for models with false predictions and 1.00 for those with correct predictions.

Confusion Plot

This is a graph that illustrates the actual and predicted observations in the models. The confusion plot has n rows and n columns with n being the total number of classes in the dataset. The confusion plot guarantees efficiency in the values and can be used to calculate metrics such as precision, recall, and accuracy.

Accuracy

The accuracy metric was also used to conduct the study. The metric is commonly used to assess model's performance and can be determined by dividing the number of correct predictions by the total number of observations. Accuracy is expressed as a percentage and the value ranges from 0 to 100. The formula is shown below

$$\text{Accuracy} = \frac{\text{True Positives + True Negatives}}{\text{All Samples}}$$

Figure 2: Accuracy

Precision

The precision metric, also known as the positive predictive value (PPV) is expressed as a ratio of the positive classes that were correctly predicted against the total predictions. The formula for precision is shown below

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Figure 3: Precision

Recall

Recall determines the correctly predicted classes based on the total number of observations. It is also referred to as sensitivity. The formula for recall is as shown below

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

Figure 4: Recall

IV. DESIGN SPECIFICATION

DATA REPOSITORY

The selected data (IoT23) was produced during a research by [41]. This network traffic dataset was first made publicly available in January 2020. A subset of this dataset was used to train and test the proposed NIDS model. The data comprised of 65107642 network captures with 21 network features. The diagram below shows the distribution of the network type present in the dataset.

```
In [8]: #Types and number of each traffic represented in the dataset
        df['tunnel_parents   label   detailed-label'].value_counts()
```

```
Out[8]: -   Malicious   PartOfAHorizontalPortScan          27311187
        -   Malicious   DDoS                               15840474
        -   Malicious   Okiru                              13655215
        -   Benign   -                                      8293827
        -   Malicious   C&C-HeartBeat                         6834
        -   Malicious   C&C                                     81
        -   Malicious   C&C-FileDownload                        12
        -   Malicious   Attack                                   7
        -   Malicious   PartOfAHorizontalPortScan-Attack         5
        Name: tunnel_parents   label   detailed-label, dtype: int64
```

Port scan, DDoS, Okiru and Benign made up the top 4 distributions of the network captures, consequently, these 4-network traffic were chosen for this project.

Additionally, a description of the 21 network features is show below.

Field	Type	Description
ts	time	Timestamp
uid	string	Unique ID of Connection
id.orig_h	addr	Originating endpoint's IP address (AKA ORIG)
id.orig_p	port	Originating endpoint's TCP/UDP port (or ICMP code)
id.resp_h	addr	Responding endpoint's IP address (AKA RESP)
id.resp_p	port	Responding endpoint's TCP/UDP port (or ICMP code)
proto	transport_proto	Transport layer protocol of connection
service	string	Dynamically detected application protocol, if any
duration	interval	Time of last packet seen – time of first packet seen
orig_bytes	count	Originator payload bytes; from sequence numbers if TCP
resp_bytes	count	Responder payload bytes; from sequence numbers if TCP
conn_state	string	Connection state (see conn.log:conn_state table)
local_orig	bool	If conn originated locally T; if remotely F. If Site::local_nets empty, always unset.
missed_bytes	count	Number of missing bytes in content gaps
history	string	Connection state history (see conn.log:history table)
orig_pkts	count	Number of ORIG packets
orig_ip_bytes	count	Number of ORIG IP bytes (via IP total_length header field)
resp_pkts	count	Number of RESP packets
resp_ip_bytes	count	Number of RESP IP bytes (via IP total_length header field)
tunnel_parents	set	If tunneled, connection UID of encapsulating parent (s)

Retrieved from, *http://doi.org/10.5281/zenodo.4743746)*

V. DESIGN SPECIFICATION

DATA PRE-PROCESSING

RELABELLING

To begin with, the name of the labels of the traffic type were adjusted to make them shorter and easier to understand. - Malicious PartOfAHorizontalPortScan' was replaced with 'PortScan', '- Malicious DDoS' was replaced with 'DDoS', '- Malicious Okiru' was replaced with 'Okiru' and '- Benign -' was replaced with 'Benign'.

```
In [10]: #Rename traffic types
         df['label'] = df['label'].replace('-   Malicious   PartOfAHorizontalPortScan', 'PortScan')
         df['label'] = df['label'].replace('-   Malicious   DDoS', 'DDoS')
         df['label'] = df['label'].replace('-   Malicious   Okiru', 'Okiru')
         df['label'] = df['label'].replace('-   Benign   -', 'Benign')
```

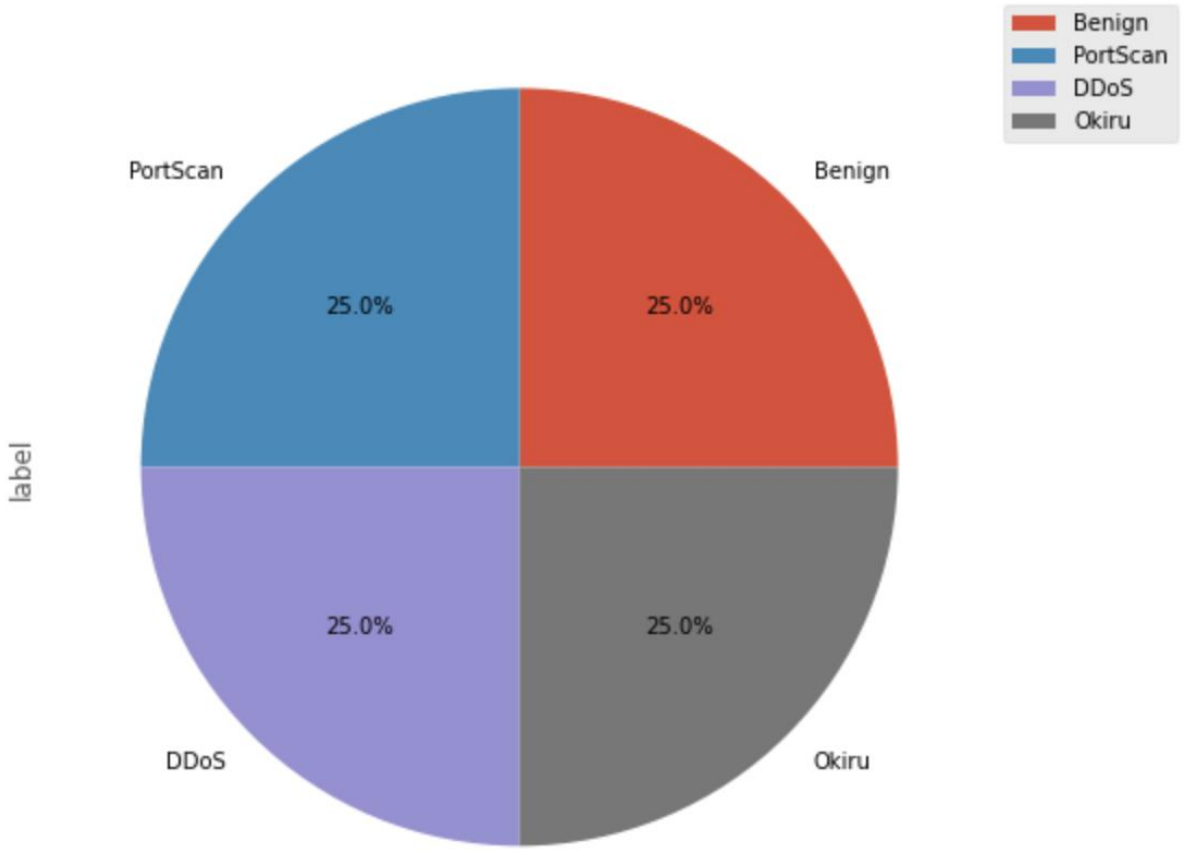
DATA BALANCING

Additionally, from the image 1 above (showing the distribution of network traffic), there is a clear imbalance in the distribution of the data. This distribution is also represented below.

S/N	Traffic Type	Distribution
1	Port Scan	27,311,187
2	DDoS	15,840,474
3	Okiru	13,655,215
4	Benign	8,293,927

Owing to limited computational power and the need for a balanced dataset to drive an effective modelling phase, 5 million samples from each traffic type is selected to for further analysis. This resulted in a balanced dataset represented below.

```
In [11]: random = 123 #Random state rs ensure reproducability of results each time the code is run
         portScan = df.query('label=="PortScan"').sample(n = 5000000, random_state = random)
         okiru = df.query('label=="Okiru"').sample(n = 5000000, random_state = random)
         benign = df.query('label=="Benign"').sample(n = 5000000, random_state = random)
         ddos = df.query('label=="DDoS"').sample(n = 5000000, random_state = random)
```



a. Null Values

A snapshot (shown below) of the data shows that missing values were represented as "-" in the dataset. In this instance "0" is not regarded as a missing value. Consequently, columns that contained only "-" as the entry were converted to numpy NaN values to make it easier to count null values in the dataset.

```
df.head()
```

	ts	uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	proto	service	duration	orig_bytes	...	conn_state	lo
0	2018-09-07 06:48:35.086226	CjKfOn3HBY0Q9XSbu4	192.168.100.111	18088	212.38.19.44	80	tcp	-	0 days 00:00:00.000002000	0	...	S0	
1	2018-09-06 19:55:08.347506	CTwFrA37B3cqIdNlpe	192.168.100.111	18088	212.245.120.75	80	tcp	-	0 days 00:00:00.000002000	0	...	S0	
2	2018-09-07 01:01:16.193973	CJrhp749zn0Bo92QFe	192.168.100.111	17576	4.184.224.179	8081	tcp	-	0 days 00:00:00.000002000	0	...	S0	
3	2018-09-07 01:01:38.448529	CZkOCF4t1R8RT6GMh9	192.168.100.111	17576	212.191.14.110	8081	tcp	-	0 days 00:00:00.000002000	0	...	S0	
4	2018-09-07 01:15:03.956982	COad3U3ND2EjUA5nGg	192.168.100.111	18088	35.248.6.170	80	tcp	-	0 days 00:00:00.000002000	0	...	S0	

5 rows × 21 columns

Following this, the number of missing values in each column are counted and represented below.

Checking for Null values

```
df.isnull().sum()
```

```
id.orig_h               0
id.orig_p               0
id.resp_h               0
id.resp_p               0
proto                   0
service          19991921
duration                0
orig_bytes              0
resp_bytes              0
conn_state              0
local_orig       20000000
local_resp       20000000
missed_bytes            0
history                 0
orig_pkts               0
orig_ip_bytes           0
resp_pkts               0
resp_ip_bytes           0
label                   0
dtype: int64
```

This shows that 3 columns, (service, local_orig and local_resp), have majority of their columns as missing values. Thus these 3 columns can be discarded as they would not contribute any information to the model.

b. Variable Encoding

Finally, the columns which are non-numerical data types are encoded and represented with numerical values to allow the machine learning and deep learning algorithms to easily interpret the data.

Variable Encoding

In [9]:
```
df["id.orig_h"] = df["id.orig_h"].astype('category').cat.codes
df["id.resp_h"] = df["id.resp_h"].astype('category').cat.codes
df["proto"] = df["proto"].astype('category').cat.codes
df["duration"] = df["duration"].astype('category').cat.codes
df["orig_bytes"] = df["orig_bytes"].astype('category').cat.codes
df["resp_bytes"] = df["resp_bytes"].astype('category').cat.codes
df["conn_state"] = df["conn_state"].astype('category').cat.codes
df["history"] = df["history"].astype('category').cat.codes
df["label"] = df["label"].astype('category').cat.codes
```

Feature Selection

The choice of feature selection technique here is the Univariate Feature Selection. This uses statistical analysis, in this case, the chi square test, to select the top features that best describe each traffic type. The image below shows the top 10 best features identified using the Univariate Feature Selection.

	Features	Score
2	id.resp_h	1.135168e+12
5	duration	1.041933e+12
3	id.resp_p	3.615433e+11
1	id.orig_p	1.161968e+11
6	orig_bytes	4.343725e+08
12	orig_ip_bytes	8.281666e+07
15	label	1.666667e+07
11	orig_pkts	2.448942e+06
14	resp_ip_bytes	2.378789e+06
7	resp_bytes	7.474695e+05

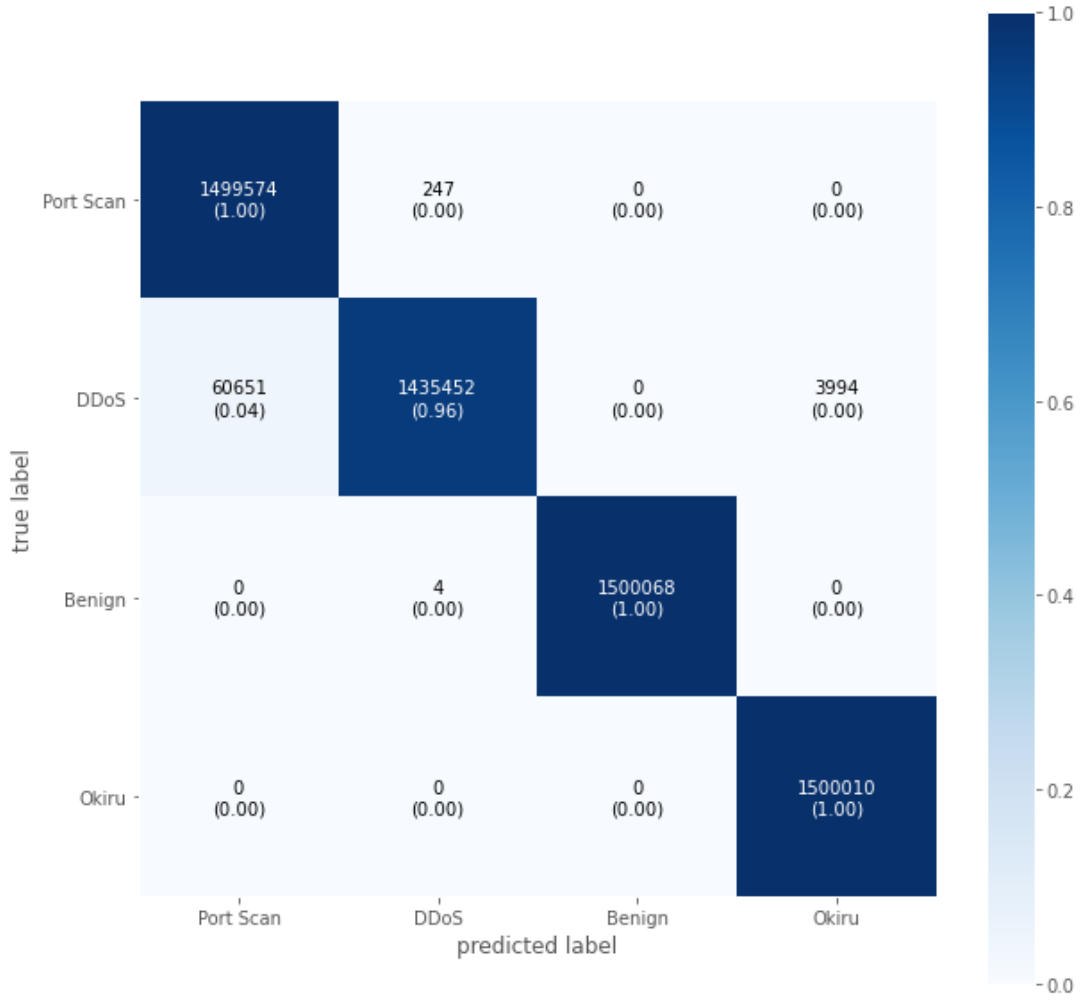
The features are assigned scores and are arranged according to their scores. The scores show how important each feature is at identifying the network traffic type.

VI. EVALUATION

This section discusses the results of the experiments conducted in this study. The study will also provide a general overview of the performance evaluation based on the algorithms that were earlier identified. The confusion matrix was used to visualise the correctly predicted classifications and falsely predicted classifications across each traffic type. Additionally, each model (experiment) was evaluated based on the precision, recall, f1-score and accuracy. The figures below demonstrate the confusion matrix for running the three experiments.

The algorithms used to train the models were the support vector machine, random forest and neural network. From the training and validation loss and accuracy chart, the model does not appear to be over fitting. Each model was then tested with different data (test data) which was not used in the training process. The test data also includes the 4 categories of traffic in the main data.

Experiment 1: **Support Vector Machine Classifier**

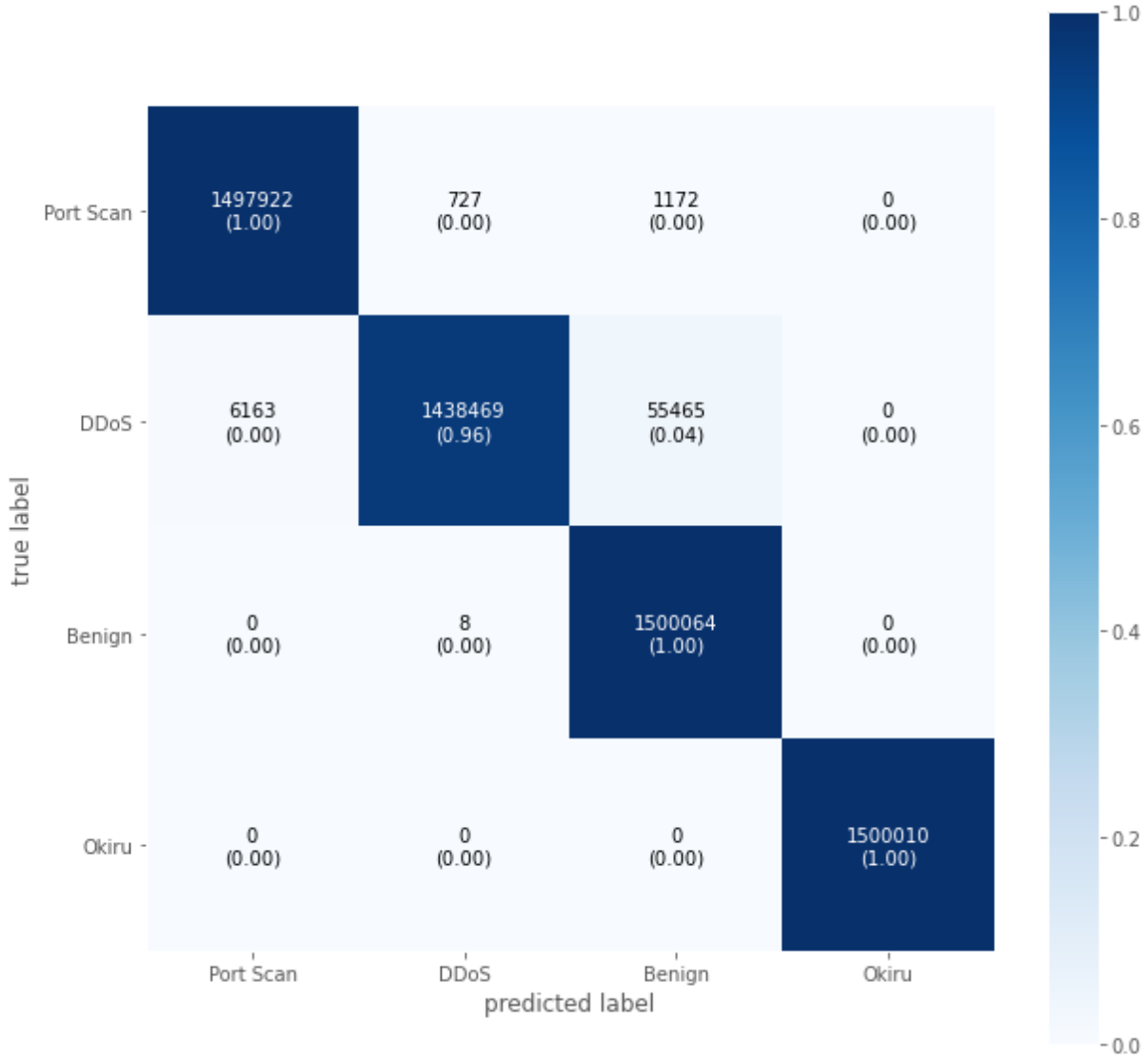


The results of the SVM algorithm using the percentage split technique

	precision	recall	f1-score	support
Port Scan	0.9611	0.9998	0.9801	1499821
DDoS	0.9998	0.9569	0.9779	1500097
Benign	1.0000	1.0000	1.0000	1500072
Okiru	0.9973	1.0000	0.9987	1500010
accuracy			0.9892	6000000
macro avg	0.9896	0.9892	0.9892	6000000
weighted avg	0.9896	0.9892	0.9892	6000000

Experiment 2: Random Forest Classifier

Confusion Matrix for RFC

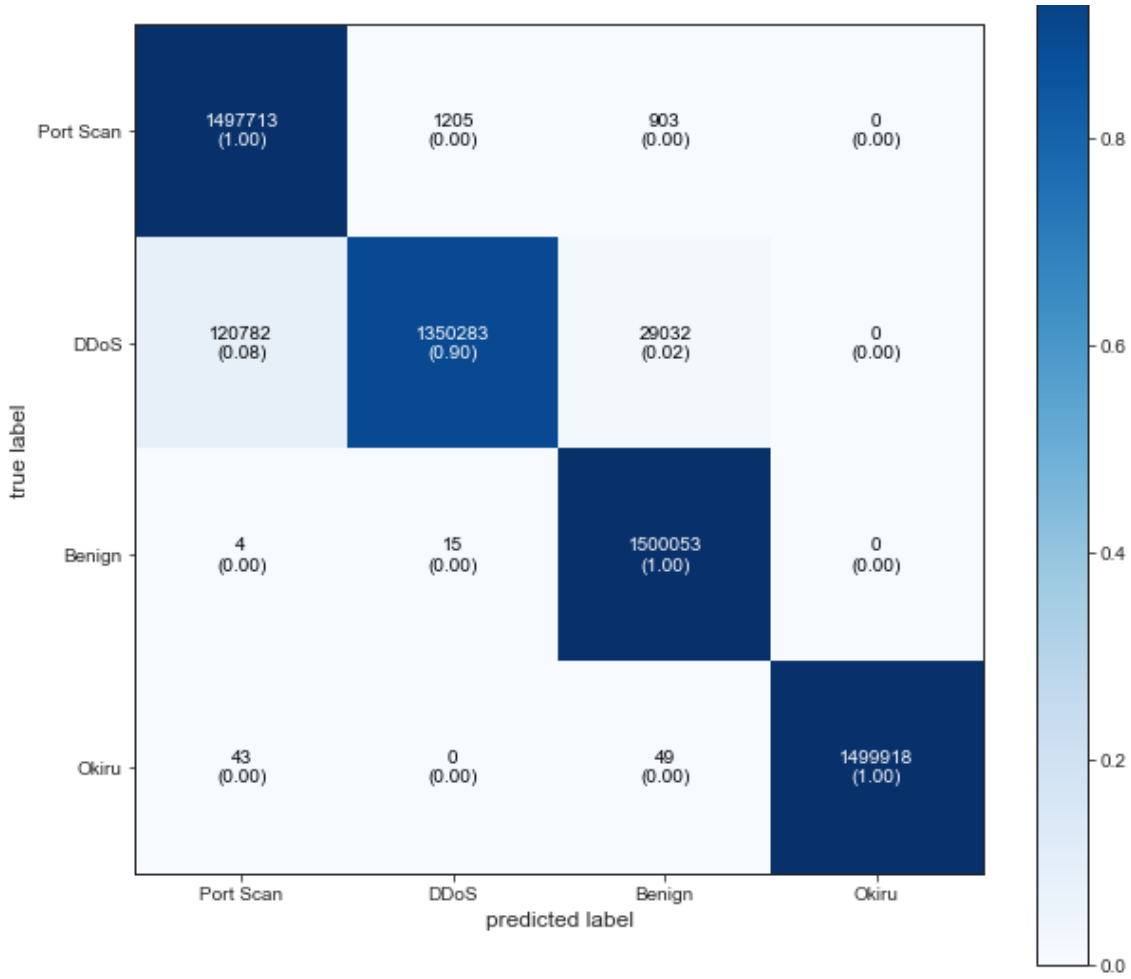


The results of the Random Forest Classifier algorithm using the percentage split technique

	precision	recall	f1-score	support
Port Scan	0.9959	0.9987	0.9973	1499821
DDoS	0.9995	0.9589	0.9788	1500097
Benign	0.9636	1.0000	0.9815	1500072
Okiru	1.0000	1.0000	1.0000	1500010
accuracy			0.9894	6000000
macro avg	0.9898	0.9894	0.9894	6000000
weighted avg	0.9898	0.9894	0.9894	6000000

Experiment 3: Neural Network

Confusion Matrix for Neural Network Algorithm



The results of the Neural Network algorithm using the percentage split technique

	precision	recall	f1-score	support
Port Scan	0.9253	0.9986	0.9606	1499821
DDoS	0.9991	0.9001	0.9470	1500097
Benign	0.9804	1.0000	0.9901	1500072
Okiru	1.0000	0.9999	1.0000	1500010
accuracy			0.9747	6000000
macro avg	0.9762	0.9747	0.9744	6000000
weighted avg	0.9762	0.9747	0.9744	6000000

VII. DISCUSSION

This section discusses the results of the study based on the experiments described above. The classification of the datasets for this study was done based on the algorithms that were mentioned in the earlier sections (SVM, Random Forest, and Neural Network). The percentage split technique which is an analysis approach used to verify the data mining model will be used for the classification. The tool works on the basis of division whereby the dataset was divided into different folds and each piece used to test the remaining sample. Likewise, the study split the dataset into 70% training and 30% for testing using the percentage split. Dividing the dataset into different pieces is necessary to ensure the correct portion of class value for every fold.

The Random Forest Algorithm together with the support vector machine classifier were run on datasets using the percentage split technique. When a comparison of the output of the machine learning algorithms was done, it was determined that the support vector machine classifier and the random vector variable produced almost similar results. The accuracy score for the SVMC was at 0.9894 while that of the RFC was 0.9892. while the values for these two algorithms are almost similar, there was a relatively significant disparity in the accuracy level for the deep learning algorithms. In part, the test on neural network algorithm recorded an accuracy of 0.9747 which compared to the machine learning algorithms, is a significant difference. The classification report for the neural network compared to the machine learning algorithms also show noticeable difference.

Conclusion and Future Work

This study aimed answering the research question: "how well can machine learning and deep learning algorithms detect network intrusion attacks?" In order to achieve this, three experiments were performed based on three different algorithms; Randon Forest Classifier, support vector machine classifier, and the neural network algorithm. The third experiment involving a deep learning algorithm, neural network algorithm produced the least performance in comparison to the other two. The first and second experiments which involved machine learning algorithms both produced almost similar results.

To conclude, this research emphasises the need to improve information security measures given the high rate of intrusion reported in recent times. Some of the constraints to take into consideration would be factors such as limited power, storage, and processing capabilities before training a dataset to mitigate potential network threats. These constraints also qualify as some of the challenges that were encountered while conducting these experiments. In addition, only a small percentage of videos from the original dataset was used thus, accuracy may not be guaranteed. To mitigate these challenges and improve accuracy in future, a different approach to conducting the experiments could be adopted. This would involve using a complete dataset instead of just a fraction of it. Future research could focus on determining how other machine learning and deep learning algorithms such as decision tree, logistic regression, and recurrent neural network could detect network intrusion.

VIII. REFERENCES

[1] R. G. Bace and P. Mell, *Intrusion detection systems*. US Department of Commerce, Technology Administration, National Institute of …, 2001.

[2] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, 'Intrusion detection by machine learning: A review', *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994–12000, Dec. 2009, doi: 10.1016/j.eswa.2009.05.029.

[3] D. Evans, 'The internet of things: How the next evolution of the internet is changing everything', *CISCO white paper*, vol. 1, no. 2011, pp. 1–11, 2011.

[4] E. Hodo *et al.*, 'Threat analysis of IoT networks using artificial neural network intrusion detection system', in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, Yasmine Hammamet, Tunisia, May 2016, pp. 1–6, doi: 10.1109/ISNCC.2016.7746067.

[5] K.-K. R. Choo, 'The cyber threat landscape: Challenges and future research directions', *Computers & security*, vol. 30, no. 8, pp. 719–731, 2011.

[6] N. Hoque, M. H. Bhuyan, R. C. Baishya, D. K. Bhattacharyya, and J. K. Kalita, 'Network attacks: Taxonomy, tools and systems', *Journal of Network and Computer Applications*, vol. 40, pp. 307–324, 2014.

[7] A. D. Wood and J. A. Stankovic, 'Denial of service in sensor networks', *computer*, vol. 35, no. 10, pp. 54–62, 2002.

[8] G. Bruneau, 'The history and evolution of intrusion detection', *SANS Institute*, vol. 1, 2001.

[9] R. A. Kemmerer and G. Vigna, 'Intrusion detection: a brief history and overview', *Computer*, vol. 35, no. 4, pp. supl27–supl30, 2002.

[10] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, 'Intrusion detection system: A comprehensive review', *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, Jan. 2013, doi: 10.1016/j.jnca.2012.09.004.

[11] P. Stavroulakis and M. Stamp, *Handbook of information and communication security*. Springer Science & Business Media, 2010.

[12] G. Xiang, H. Jin, D. Zou, X. Zhang, S. Wen, and F. Zhao, 'VMDriver: A driver-based monitoring mechanism for virtualization', in *2010 29th IEEE Symposium on Reliable Distributed Systems*, 2010, pp. 72–81.

[13] 'Anomaly-Based Detection - an overview | ScienceDirect Topics'. https://www.sciencedirect.com/topics/computer-science/anomaly-based-detection (accessed Feb. 26, 2021).

[14] C. Kruegel and T. Toth, 'Using decision trees to improve signature-based intrusion detection', in *International Workshop on Recent Advances in Intrusion Detection*, 2003, pp. 173–191.

[15] H. Liu and B. Lang, 'Machine learning and deep learning methods for intrusion detection systems: A survey', *applied sciences*, vol. 9, no. 20, p. 4396, 2019.

[16] Y. Wu, D. Wei, and J. Feng, 'Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey', *Security and Communication Networks*, vol. 2020, pp. 1–17, Aug. 2020, doi: 10.1155/2020/8872923.

[17] R. Sathya and A. Abraham, 'Comparison of supervised and unsupervised learning algorithms for pattern classification', *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, no. 2, pp. 34–38, 2013.

[18] L. Koc, T. A. Mazzuchi, and S. Sarkani, 'A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier', *Expert Systems with Applications*, vol. 39, no. 18, pp. 13492–13500, 2012.

[19] M. Panda and M. R. Patra, 'Network intrusion detection using naive bayes', *International journal of computer science and network security*, vol. 7, no. 12, pp. 258–263, 2007.

[20]Y. Yi, J. Wu, and W. Xu, 'Incremental SVM based on reserved set for network intrusion detection', *Expert Systems with Applications*, vol. 38, no. 6, pp. 7698–7707, Jun. 2011, doi: 10.1016/j.eswa.2010.12.141.

[21]G. L. Grinblat, L. C. Uzal, and P. M. Granitto, 'Abrupt change detection with one-class time-adaptive support vector machines', *expert systems with applications*, vol. 40, no. 18, pp. 7242–7249, 2013.

[22]Y.-T. Hou, Y. Chang, T. Chen, C.-S. Laih, and C.-M. Chen, 'Malicious web content detection by machine learning', *expert systems with applications*, vol. 37, no. 1, pp. 55–60, 2010.

[23]H.-C. Wu and S.-H. S. Huang, 'Neural networks-based detection of stepping-stone intrusion', *expert systems with applications*, vol. 37, no. 2, pp. 1431–1437, 2010.

[24]C. Elkan, 'The foundations of cost-sensitive learning', in *International joint conference on artificial intelligence*, 2001, vol. 17, no. 1, pp. 973–978.

[25]G. M. Weiss, K. McCarthy, and B. Zabar, 'Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?', *Dmin*, vol. 7, no. 35–41, p. 24, 2007.

[26]J. Barnard and X.-L. Meng, 'Applications of multiple imputation in medical studies: from AIDS to NHANES', *Statistical methods in medical research*, vol. 8, no. 1, pp. 17–36, 1999.

[27]J. Kaiser, 'Dealing with Missing Values in Data', *JoSI*, pp. 42–51, 2014, doi: 10.20470/jsi.v5i1.178.

[28]J. Tang, S. Alelyani, and H. Liu, 'Feature selection for classification: A review', *Data classification: Algorithms and applications*, p. 37, 2014.

[29] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, 2020, doi: 10.1002/ett.4150.

[30] A. Pathak and S. Pathak, "Study on decision tree and knn algorithm for intrusion detection system," *International Journal of Engineering Research and Technology (IJERT)*, vol. 9, no. 5, 2020, doi: 10.17577/ijertv9is050303.

[31] E. M. Roopa Devi and R. C. Suganthe, "Enhanced transductive support vector machine classification with grey wolf optimizer cuckoo search optimization for intrusion detection system," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 4, 2018, doi: 10.1002/cpe.4999.

[32] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," *BMC Medical Informatics and Decision Making*, vol. 19, no. 1, 2019, doi: 10.1186/s12911-019-1004-8.

[33] A. O. Alzahrani and M. J. Alenazi, “Designing a network intrusion detection system based on machine learning for software defined networks,” *Future Internet*, vol. 13, no. 5, pp. 111–128, 2021, doi: 10.3390/fi13050111.

[34] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, vol. 7, pp. 41525–41550, 2019, doi: 10.1109/ACCESS.2019.2895334.

[35] F. Aldakheel, R. Satari, and P. Wriggers, “Feed-forward neural networks for failure mechanics problems,” *Applied Sciences*, vol. 11, no. 14, pp. 6483–6504, 2021, doi: 10.3390/app11146483.

[36] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[37] L. Jin, F. Tan, and S. Jiang, “Generative adversarial network technologies and applications in computer vision,” *Computational Intelligence and Neuroscience*, vol. 2020, pp. 1–17, 2020, doi: 10.1155/2020/1459107.

[38] C. Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks,” *IEEE Access*, vol. 5, pp. 21954–21961, 2017, doi: 10.1109/access.2017.2762418.

[39] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018, doi: 10.1109/tetci.2017.2772792.

[40] D. Papamartzivanos, F. Gomez Marmol, and G. Kambourakis, “Introducing deep learning self-adaptive misuse network intrusion detection systems,” *IEEE Access*, vol. 7, pp. 13546–13560, 2019, doi: 10.1109/access.2019.2893871.

[41] A. Parmisano, S. Garcia, and M. J., Erquiaga. "A labeled dataset with malicious and benign iot network traffic." *Stratosphere Laboratory: Praha, Czech Republic* (2020).