INTRUSION DETECTION USING BIG DATA AND DEEP LEARNING TECHNIQUES

OSAMA MOHAMED FAKER FAKER

January, 2019

INTRUSION DETECTION USING BIG DATA AND DEEP LEARNING TECHNIQUES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL
AND APPLIED SCIENCES
OF ÇANKAYA UNIVERSITY

BY

Osama Mohamed Faker FAKER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JANUARY 2019

Title of the Thesis: **INTRUSION DETECTION USING BIG DATA AND DEEP LEARNING TECHNIQUES**

Submitted by: **Osama Mohamed Faker FAKER**

Approval of the Graduate School of Natural and Applied Sciences, Cankaya University.

Prof. Dr.Can ÇOĞUN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr.Erdoğan DOĞDU
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr.Erdoğan DOĞDU
Supervisor

Examination Date:21/1/2019

Examining Committee Members:

Prof. Dr. Erdoğan DOĞDU	(Çankaya Univ.)
Dr. Öğr. Üyesi Ziya Karakaya	(Atılım Univ.)
Dr. Öğr. Üyesi Roya Choupani	(Çankaya Univ.)

STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : **Osama Mohamed Faker FAKER**

Signature :

Date : **21/1/2019**

ABSTRACT

INTRUSION DETECTION USING BIG DATA

AND DEEP LEARNING TECHNIQUES

FAKER, Osama
M.Sc., Computer Engineering Department
Supervisor: Prof. Dr. Erdoğan DOGDU

2019, 80 pages

Intrusion detection systems are one of the most important systems in cybersecurity that are designed to prevent and detect attacks by analyzing network traffic to look for abnormal patterns that are likely to represent hidden attacks. Volume, velocity, and variety are the characteristics of big data and represent the great challenge of intrusion detection systems, making it difficult to monitor and analyze this large volume of data using traditional techniques. In this study, big data and deep learning techniques are integrated to improve the performance of intrusion detection systems. Three classifiers are used to classify the network traffic datasets, and those are Deep Feed-Forward Neural Network and two ensemble techniques, Random Forest and Gradient Boosting Tree. To select the most relevant attributes from the datasets, we use a homogeneity metric to evaluate features. Two recently published datasets UNSW-NB15 and CICIDS2017 are used to evaluate the proposed method. 5-fold cross validation is used in this work to evaluate the machine learning model. We implemented the method using the distributed computing environment Apache Spark, integrated with Keras Deep Learning Library to implement the deep learning technique while the ensemble techniques are implemented using Apache Spark Machine Learning Library. The results show a high accuracy with DNN for binary and multiclass classification on UNSW-NB15 dataset with very short prediction time and with accuracies at 99.16% for binary classification and 97.01% for multiclass classification. While the GBT classifier achieved the best accuracy for binary classification with the CICIDS2017 dataset, which at 99.99% and for multiclass classification the DNN accuracy the highest with 99.56%.

Keywords: Intrusion detection system, big data, machine learning, artificial neural networks, deep learning, ensemble techniques, feature selection.

ÖZ

BÜYÜK VERİ VE DERİN ÖĞRENME
TEKNİKLERİNİ KULLANARAK SALDIRI TESPİTİ

FAKER, Osama

Yüksek Lisans, Bilgisayar Mühendisliği Departmanı

Danışman: Prof. Dr. Erdoğan DOGDU

2019, 80 sayfa

Saldırı tespit sistemleri, gizli saldırıları temsil etmesi muhtemel anormal kalıpları aramak için ağ trafiğini analiz ederek saldırıları önlemek ve tespit etmek için tasarlanmış siber güvenlik alanındaki en önemli sistemlerden biridir. Hacim, hız ve çeşitlilik büyük verilerin özellikleridir. Geleneksel teknikleri kullanarak, büyük veri hacminin izlenmesini ve analiz edilmesini zorlaştıracak olan saldırı tespit sistemleri büyük zorluğunu temsil etmektedir. Bu çalışmada, büyük veri ve derin öğrenme teknikleri, izinsiz giriş tespit sistemlerinin performansını iyileştirmek için entegre edilmiştir. Ağ trafiği veri kümelerini sınıflandırmak için üç sınıflandırıcı kullanılır. Bunlar; Derin Ileri Beslemeli Yapay Sinir Ağı (Deep Feed-Forward Neural Network) ve iki ensemble öğrenme tekniğidir; bunlar da Rastgele Orman (Random Forest) ve Gradyan Artırma Ağacı (Gradient Boosting Tree). Veri kümelerinden en ilgili özellikleri seçmek ve bunları değerlendirmek için homojenlik ölçümünü kullanıyoruz. Yeni yayınlanan iki veri seti UNSW-NB15 ve CICIDS2017, önerilen yöntemi değerlendirmek için kullanılmıştır. Makine öğrenim modelini değerlendirmek için bu çalışmada 5 kat çapraz doğrulama kullanılmıştır. Apache Spark Machine Learning Library(Apache Spark Makine Öğrenme Kütüphanesi) kullanılarak ensemble teknikleri uygulanırken, derin öğrenme tekniğini uygulamak için Keras Deep Learning Library (Keras Derin Öğrenme Kütüphanesi) ile entegre olan dağıtılmış bilgi işlem ortamı Apache Spark’ı kullanarak bu yöntemi uyguladık. Sonuçlar, DNS'nin UNSW-NB15 veri setinde ikili ve çoklu sınıf sınıflandırması için yüksek bir hassasiyet olduğunu ve çok kısa bir süre için öngörülen sürenin, ikili sınıflandırma için %99,16, çoklu sınıflandırma için %97,01 olduğunu ve GBT sınıflandırıcısının en iyi ikili sınıflandırma doğruluğunu elde ettiğini gösterdi. %99.99 olan CICIDS 2017 veri seti ve çok sınıflı sınıflandırma için DNN doğruluğu %99.56 ile en yüksek seviyedeydi.

Anahtar Kelimeler: Saldırı tespit sistemi, büyük veri, makine öğrenmesi, yapay sinir ağları, derin öğrenme, topluluk teknikleri, özellik seçimi.

ACKNOWLEDGMENTS

Foremost, I would like to thank my god ALLAH who support me to achieve the success. I would like to express my sincere gratitude to my advisor Prof. Dr. Erdoğan DOGDU for the continuous support of my thesis, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

It is a pleasure to express my very profound gratitude to my family especially my parents and my wife for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Lastly, I would like to thank all the staff in my department at Çankaya University who supported me throughout academic courses.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IDS	: Intrusion Detection System
ML	: Machine Learning
ANN	: Artificial Neural Network
DL	: Deep Learning
RF	: Random Forest
GBT	: Gradient Boosting Tree
TanH	: Hyperbolic Tangent
ReLU	: Rectified linear unit
CPU	: Central Processing Unit
GPU	: Graphic Processing Unit

CHAPTER 1

INTRODUCTION

Providing protection and privacy of big data is one of the most important challenges facing developers of security management systems. Especially with the large expansion of the use of Internet networks and the rapid growth of the volume of data generated from several sources. This expansion and growth gave more space for hackers to launch their malicious attacks and development techniques and tools of intrusion. On the other hand, researchers and developers of intrusion detection systems (IDSs) seek to increase the efficiency of detecting malicious attacks and predicting them early. Intrusion detection systems are one of the most important systems used in cybersecurity.

Intrusion refers to attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network resources. Intrusion detection systems (IDSs) are the hardware or software that monitors and analyzes data flowing through computers and networks to detect security breaches that threaten confidentiality, integrity or availability of a system's resources [1]. There are three types of intrusion detection system, Host-based IDS, Network-based IDS, and Hybrid-based IDS, depending on the IDS’ location in the network. Host-based IDS monitors and collects data flowing within a computer or host and with the large growth of networks. The need for network-based IDS has emerged, which focus on monitoring the flow of data within a specific network or network group. The integration between the Host-based IDS and Network-based IDS created a more efficient system to monitor the flow of data within the devices and across the network, which is called Hybrid-based IDS [2].

Intrusion detection systems are using two basic methods for analyzing events to detect attacks: (I) misuse detection and (ii) anomaly detection. Misuse detection or signature-based detection is an analysis of system activities to search and detect patterns of attacks identical or a like to previously known attack patterns and stored in a database intrusion detection system. Anomaly detection is to detect unusual patterns of behavior in network traffic and relies on building models that represent the normal behavior of users, hosts or the network where patterns of behavior that deviate from these models are detected and often represent abnormal behavior [3].

The process of anomalies detecting was classified into two ways. The first method is Programmed model, where the system is taught to detect abnormal activities by an external user. Which determines the abnormal behavior in the system, that poses a threat to system integration and security. This Programmed model has been classified into three categories: threshold, Simple rule-based and Statistical model. The second method is self-learning; this method is based on building models of the basic processes of the system by monitoring traffic for a long time. Self-learning systems are classified into two categories: time series and machine learning [4].

Big data is data that are difficult to be stored, managed, or manipulated by traditional techniques. The big data characteristics are volume, variety, and velocity, which is called 3Vs of big data [5]. Where they represent a major challenge for intrusion detection systems [6]. Volume refers to the amount of data, where data generated from several different sources have exploded very dramatically over the last years. Which requires monitoring and analysis network traffic to integrate with the management and processing of big data. The Big Data is often associated with another challenge, which is variety, that indicates different data sources and therefore various data types structured, semi-structured and unstructured data. Also, variety refers to heterogeneous data, large IT infrastructure can generate a huge amount of data from many resources like many application servers, networks and workstations. Analyzing and monitoring heterogeneous data is a complex challenge and exacerbates the problems facing intrusion detection systems. The huge change in the volume and variety of data has also led to a change in the speed of data generation and streaming, which is called: velocity. Big data velocity refers to the speed of data that flows from many sources like networks, business process, human interaction with applications like social media. The speed of data flow from multiple sources is another challenge for the developers of intrusion detection systems. There is an urgent need to build intrusion detection systems capable of coping with the speed of data flow. Thus, improving the efficiency of analyzing and monitoring data traffic, detecting and preventing intrusion [7]. IBM company added another characteristic of “V” to the three previous 3Vs of big data, the fourth characteristic is the veracity that refers to the noises, biases, and abnormality in data. The higher the data quality, the more accurate the results [8]. Yuri Demchenko [9] added another “V” to the IBM 4Vs of big data, that called value, it Furthermore, Microsoft extended the 3Vs of big data to the 6Vs, which it added veracity, variability, and visibility [10]. Where the variability refers to the complexity of data or the number of variables in a data set. While the Visibility emphasizes that need have a full picture of data to make an informative decision.

Big Data framework and related technologies have been created and developed over the last few years such as Hadoop [11], Apache Spark [12], Hive [13], NoSQL [14], which can handle big data and its various characteristics of volume, velocity, and variety. Big data techniques have many advantages such as speed in receiving, storing and processing data of various types. The characteristics of big data represent a real challenge for developers of intrusion detection systems, while the advantages of big data technologies give greater efficiency to intrusion detection systems in case of integration between them.

Apache Spark is one of the most important frameworks developed over the past few years to handle the big data. An open source framework that combines an engine for distributing programs across clusters of machines with an elegant model for writing programs on top of it. Apache Spark has been developed using MapReduce technology for big data processing, also to deal with many workloads such as interactive queries and streaming, batch applications, frequent algorithms. The most important feature of Apache Spark is cluster computing in memory. It supports and provides many APIs where applications can be written in different languages (Java, Scala, Python), and it supports data flow, machine learning, SQL queries, and graph algorithms [15]. Machine learning library provided by Apache Spark which includes many machine learning algorithms that are widely used in many fields such as anomaly detection, natural language processing and recommendation engines etc. Machine learning algorithms are divided into many categories according to the tasks performed by the algorithm such as classification, prediction, clustering etc. [16]. The Apache Spark integrates with a range of deep learning libraries that allow deep learning algorithms to be implemented on a Spark computing environment quickly, reducing training and testing time. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano [17].

Machine learning is an artificial intelligence application based on the idea that the machine can learn from data to build models and make decisions with very limited human intervention. Machine learning techniques have shown high performance and efficiency in extracting different types a pattern of knowledge from the environment in which these techniques are used. These techniques are divided into two main categories: unsupervised and supervised machine learning techniques. A term supervised indicates that this type of technology requires a training process so that the extracted outputs are an approach or a match to the expected outputs. While the term refers to the fact that these techniques do not require a training process and thus cannot be expected outputs. These techniques rely on their capability to extract knowledge from the data [18]. Machine learning includes a range of techniques such as clustering, classification, artificial neural networks. Each of these techniques includes a set of algorithms that are widely used to develop and improve the performance of anomaly detection. Recently, artificial neural networks have been used in many applications and have provided good performance and results [19]

1.1. Problem Definition

Over the past years, the number of cyber-attacks has increased. These attacks target to threat the integrity and confidentiality of networks. With the large volume of data generated from several sources, intrusion detection systems face with major challenges. Detecting and preventing attacks before they occur requires a great ability to monitoring and analyzing the big data traffic to determine and know the features and patterns of attacks. The speed of analysis and monitor of big data traffic is not provided by traditional data management techniques. which makes intrusion detection systems not capable of handling large data characteristics. Big data techniques and intrusion detection systems are involved in the use of machine learning techniques in data analysis and pattern extraction. allowing their integration to develop the capabilities of intrusion detection systems.

1.2. Aim of the Study

The aim of this study is to integrate deep learning and big data processing techniques to improve the performance of intrusion detection systems. A hybrid approach has been proposed that includes the use of a homogeneity metric in K-means clustering algorithm to select features from datasets. Then apply a deep learning network and ensemble technique Independently with k-fold cross-validation procedure. The approach has been tested on two recent datasets that contain a set of common and modern attacks.

1.3. Thesis Layout

The rest of the thesis chapters are arranged as follow, Chapter two reviews the literature related to the techniques included in this work. Chapter three provides the proposed method in detail. The experiments conducted to evaluate the performance of the proposed method are explained in Chapter four. Chapter five discusses the results of the experiments. Finally, chapter six presents the concluding remarks of the study.

CHAPTER 2

LITERATURE REVIEW

The intrusion detection systems are different in terms of design and structure. Since the development of the first model for intrusion detection by Dorothy [20]. many intrusion detection systems have been designed and developed, which differ in their data observation and collection techniques. However, most of them rely on the basic architectural form shown in Figure 2.1. It consists of the following parts: Data gathering device (sensor), the function of this component is to collect data from a monitored system; Detector (Intrusion Detection (ID) analysis engine) that is responsible for processing data collected from the sensors to determine anomaly activity. Knowledgebase (Database) that includes data collected from sensors after preprocessing, which represent information of attack patterns and data profiles. Configuration devise that provides information about the status of the Intrusion Detection System (IDS). Response component that is responsible for initiating intrusion detection procedures [21].

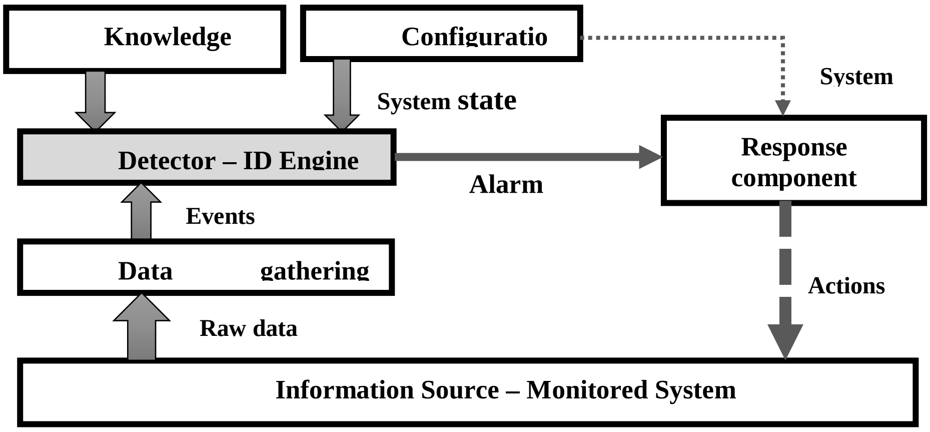

Figure 2.1: Basic architectural design of an Intrusion Detection System [21]

The intrusion detection systems are based on two main methods of detection. One of them is misuse-based detection, which is also called signature-based detection. This method is based on a database that includes attack patterns and characteristics. Where this technique monitors and analyzes data to detect any data that matches the patterns or characteristics of items in the database. On the other hand, the other method, which is called anomaly-based detection, attempts to estimate the normal behavior of the system that needs to be protected from abnormal patterns. that are detected and deviate from the normal behavior of the system. Anomaly-based detection is to detect unusual patterns of behavior in network traffic and relies on building models that represent the normal behavior of users, hosts, or the network. Where patterns of behavior that deviate from these models are detected and often represent abnormal behavior. As in the following Figure 2.2, which illustrates the anomaly in a two-dimensional data set, where groups N1 and N2 are normal data sets because they contain most of the data points, while the data points in the A3 set and A4 and A5 are considered as the anomalies because they are so far away from the two normal assumed datasets N1 and N2 [22].

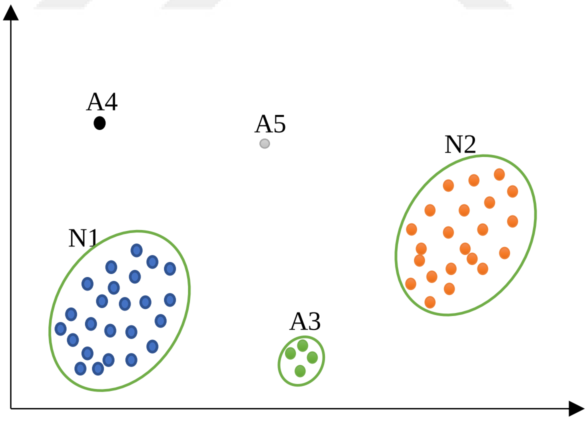

Figure 2.2: A simple example of anomalies [22]

As in Figure 2.2, anomaly-based detection is highly dependent on multiple machine learning techniques. That have the ability to extract patterns in data and build models based on a set of features that can classify the abnormal behavior of data traffic to and from the network or devices. Classification, clustering, shallow and deep neural networks are some of the most widely used machine learning techniques in developing and improving the performance of anomaly detection systems.

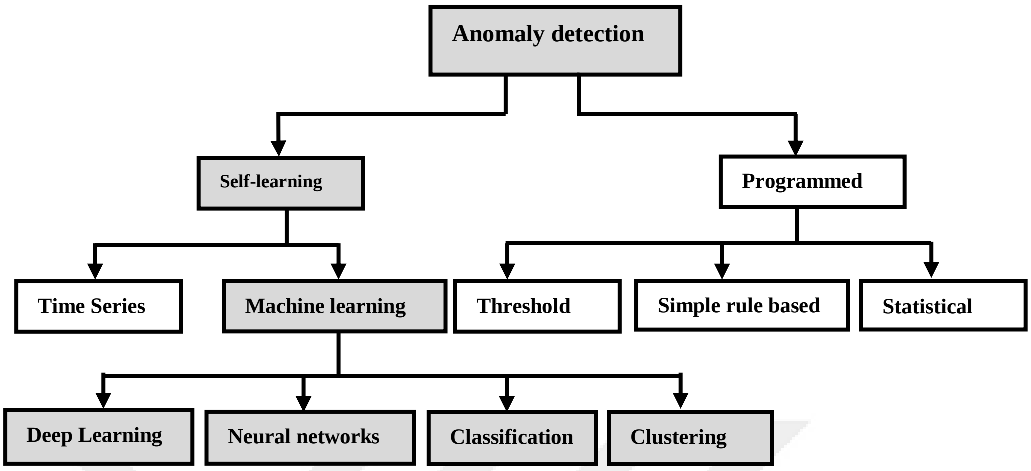

Figure 2.3: Techniques used in anomaly detection

Several previous and modern works discussed the techniques of anomaly-based detection and their applications and approaches based on machine learning techniques. Bhuyan et al. [23] provides an overview of the research and studies, discussing the infrastructure of anomalies detection techniques. As well as the different species developed from these infrastructures, and the use of anomaly detection in many applications and challenges is also discussed. Agrawal and Agrawal [24] review various mining techniques and their approaches to detecting anomalies such as classification, clustering and hybrid techniques. Jyothsna et al. [25] this work provides a detailed survey of four major methods of anomaly detection techniques that are widely used, including classification, statistics, information theory and aggregation. Also discusses research difficulties and challenges with data sets used to detect intrusion.

The basic processes of machine learning techniques can be described in the detection of anomalies as in Figure 2.4. After collecting the required data, they are categorized and tabulated in a way that allows the application of machine learning techniques. the type of machine learning technique is determined by the way it operates supervised or unsupervised. Decision tree, K-Nearest Neighbors, neural networks are considering supervised techniques. K-means Clustering Example of unsupervised techniques.

Outputs are evaluated in two ways: scores or binary/label. Scoring-based anomaly detection techniques determined anomaly value for each data instance. While the binary- based anomaly defined the output in binary format as either anomaly or normal [26].

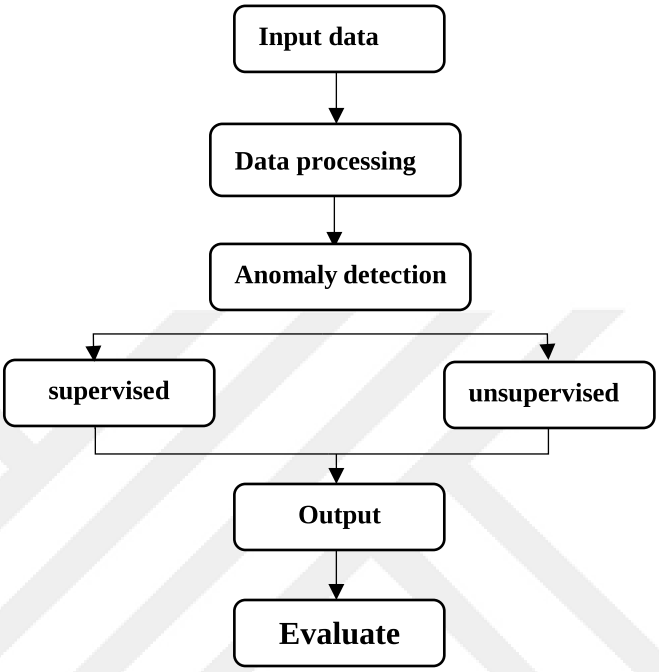

Figure 2.4: Basic processes of Machine learning in Anomaly Detection [26]

The applied of machine learning techniques to the data set is often preceded by the process of features selection. Which is an important technique for improving the performance of machine learning techniques. building highly efficient systems for monitoring and analyzing network traffic, early detection and prevention of attacks. Especially with the enormous volume of data transmitted across networks, this data is variety and has multiple sources and often associated with a high level of noise and duplicate data. feature selection is one of the common ways to get rid of noisy and irrelevant data [27].

Feature selection is a set of techniques that are used to select a subset of related features to build a high-expectation model. Datasets contain several features, some features are often more important in building a predictive model while having unneeded, irrelevant or duplicate features are likely to adversely affect the construction of a highly accurate model. Feature selection is one of the most important techniques in machine learning, which aims to improve the efficiency and performance of the predictors to obtain a high accuracy rate. Also feature selection helps to build a faster and more cost-effective predictors and provides a clearer understanding of the basic process of generating the data set [28].

There are many algorithms used to feature selection, that are divided into three main categories: filter method, wrapper method and embedded method. The filter method is based on the applies of statistical measures to determine the scoring of each feature, then features are ranked by the score, which attributes are retained, and which ones are deleted. While the wrapper method depends on the selected subset of the features and evaluation of the selected subset by the performance of classifier. Each time repeated the subset selection of the features and evaluated until the highest rating is obtained and considered as the final features set to learn the classifier then evaluated the classifier on independent testing dataset. feature selecting while building the model may produce more accurate and unbiased results, this is called an embedded method where the learning process cannot be separated from the feature selection process [29].

Hindy et al. [30] this work provided an overview of the overall classification of intrusion detection systems and their classification with recent and previous work. This classification gives a detailed description of the intrusion detection system and its complexity. It also provides an overview of feature selection techniques that affect the effectiveness of machine learning techniques and their role in training and testing.

The proposed method suggested that the homogeneity metric is used as an unsupervised feature selection for filter method. which is one of the measures used in the clustering techniques, where the homogeneity ranking of each feature is measured individually and the features with the highest rating are selected, that will be used to build the prediction model. The k-means clustering algorithm is one of the most popular clustering algorithms and is effective in handling large data. The homogeneity metric is one measure of the k mean algorithm, which indicates the extent to which a set of data points belongs to a specific class [31].

Data cluster is the process of placing data points in similar clusters, a branch of data mining. The clustering algorithm divides data points set into several groups, since the similarity between points within a particular cluster is greater than the similarity between points within two different clusters. The idea of data clustering is simple in nature and very close to human in its way of thinking. Whenever the amount of data is large, the human tends to summarize the huge amount of data to a small number of groups or categories, in order to facilitate the process of analysis. clustering algorithms are widely used not only to organize and classify data but to compress data and build a data order model. If can find clusters of data, a model of the problem can be built based on those clusters.

K-means clustering algorithm is one of the most common unsupervised machine learning algorithms. That is defined as a method in which data are divided into groups in a way that objects in each group share more similarity than with other objects in other groups. It is the most famous data analysis algorithms due to its outstanding mathematical performance. Where the data is collected in (K) cluster, (k) is the number of clusters determined before starting the algorithm. The clustering is done by reducing the total square distances (Euclidean spaces) between the elements and the center of the corresponding cluster. K is the number of clusters that is determined by the programmer. After determining the number of clusters, the Centroid is chosen randomly for each cluster. The distance between the Centroid and the data points is measured by the Euclidean equation.

$$d_{ij}\sqrt{\sum_{k=1}^{n}(x_{ik}} - x_{jk})^2 \text{..}(2.1)$$

n = data points number

x_{ik} coordinate k of data point i

x_{jk} coordinate k of data point j

The data points closest to the Centroid is grouped, the mean distance between these data points is calculated and the mean is defined as a new Centroid. The process is repeated from the second stage so that no data points moves between the clusters. Homogeneity is a clustering metric that is used to determine the homogeneity of data points in a single cluster, where clustering must assign only those data points that are members of a single class to a single cluster. That means the cluster entropy is zero, entropy refers to the randomness or unpredictability.

Define homogeneity as:

$$h = \frac{H(C,K)}{H(C)} \text{...} (2.2)$$

where $H(C,K)$ is the conditional entropy of the classes given the cluster assignments and is given by:

$$H(C,K) = -\sum_{K=1}^{|K|} \sum_{C=1}^{|C|} \frac{n_{c,k}}{n} \log \frac{n_{c,k}}{n_k} \text{......................................} (2.3)$$

and $H(C)$ is the entropy of the classes and is given by

$$H(C) = -\sum_{C=1}^{|C|} \frac{n_c}{n} \log \frac{n_c}{n} \text{..} (2.4)$$

where n the total number of data points, n_c and n_k the number of data points respectively belonging to belonging to class c and cluster k, and $n_{c,k}$ the number of data points from class c assigned to cluster k [32, 33].

The researchers presented several proposals and studies that relied on the technique of unsupervised features selection. Some of them based on one technique and others introduced a hybrid approach combining sets of techniques for feature selection and machine learning techniques. Nisioti et al. [34] This paper provides an overview of the works that have provided hybrid and unsupervised intrusion detection systems, also the techniques used and their efficiency in detecting modern attack types based on a subset of features, In addition, the study shows that unsupervised techniques have the ability to select a subset of features from data generated from different and heterogeneous sources without the need to re-training, thus reducing computational time and complexity.

Nour Mustafa and Jill Slay [35] Proposed Hybrid feature selection technique based on the central points (CP) of attribute values and Association Rule Mining (ARM). The EM clustering, Logistic Regression and Naïve Bayes are used to discriminate between attack and normal records. UNSW-NB15 & NLSKDD used to evaluate the proposed, where the Logistic Regression achieved the best accuracies on both dataset 83% of UNSW-NB15 and 82.1% of NLSKDD.

Vajiheh Hajisalem, Shahram Babaie [36] proposed a new hybrid classification method based on Artificial Bee Colony (ABC) and Artificial Fish Swarm (AFS) algorithms. The Fuzzy C-Means Clustering (FCM) and Correlation-based Feature Selection (CFS) techniques are applied to divide the training dataset and remove the irrelevant features. If-Then rules are generated through the CART technique according to the selected features to distinguish the normal and anomaly records. The proposed hybrid method was evaluated using UNSW-NB15 and NLSKDD dataset in terms of detection rate and false positive rates. Where the method applied on NLSKDD achieved 99% accuracy rate and 0.01% false positive rate, while the UNSW-NB15 achieved 98.9% accuracy rate and 0.13 false positive rate.

Mohamed Idhammad, et al. [37] Authors presented a detection approach of DoS attack based on ANN (MLP). Unsupervised correlation-based feature selection method used to select relevant features. MLP implemented with several hidden layer (1 to 10) and average of accuracy. Training time and testing time calculated for each. The best average of accuracy 0.97% achieved by the MLP with seven hidden layer that applied on UNSW-NB15 dataset, while the six hidden layers achieved 0.99% average of accuracy with NLSKDD dataset.

Artificial neural networks (ANN) are one of the main techniques used in machine learning. The word "nervous" refers to systems inspired by how the brain works to mimic the way humans learn or acquire knowledge. Neural networks are a technique created to simulate the human brain to patterns recognition, classification and clustering [38].

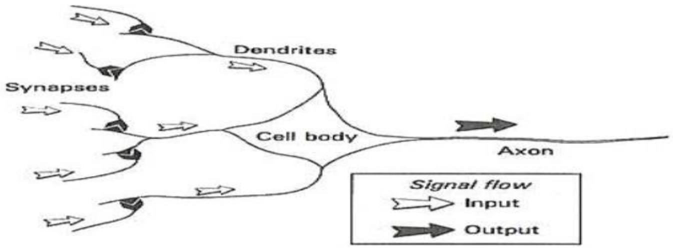

Figure 2.5: Essential components of the neuron in brain [39]

The simple neuron of the human brain consists of four parts: first, the function of Dendrite is the responsible part for receiving signals from other neurons. The Second, Soma (Cell Body) This section collects all incoming signals to generate inputs. Third, Axon Structure In the case of the total sum in the cell body to the threshold value, the signal is transmitted to other neurons by the axon. Finally, Synapses Working represents the point of connection between one neuron and the other neurons.

ANN is designed to simulate the work of basic biological neurons, where weighted inputs based on their synaptic interconnection represent the dendrites in the Biological Neural Network. Artificial neurons in the ANN, which also include a summation unit and threshold representing the cell body in biological neural networks, while the output layer in artificial neural networks performs the function of Exon.

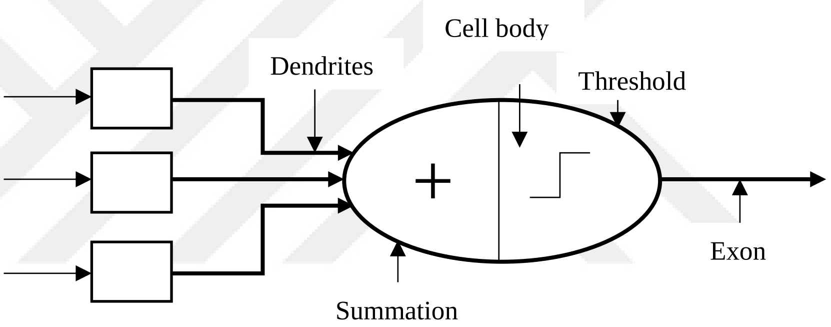

Figure 2.6: Artificial neuron Computations [40]

In artificial neural networks, to simulate the electrochemical junction in the human neurons. Adjusted the effect of the output of one neuron that moving to another neuron by multiplying the output value of that neuron by the weight between the two neurons. calculated the output of a neuron by passing the summation of the weighted inputs of that neuron through an activation function. The activation function is used to make the boundary between input values non-linear leading to greater precision in decision-making. There is many functions are used to activate the summation of the weighted inputs in a neuron, such as the rectified linear unit (ReLU), the Hyperbolic Tangent (TanH) and Sigmoid functions. The ReLU is widely used in hidden layers for their ability to improve performance and fast learning when compared to other activation functions [41].

Different activation functions are often used in the output layer neurons than those used in hidden layer neurons as desired by neural network outputs. Where the Sigmoid activation function is used for labeling problems, while the SoftMax activation function is used when the input value is assigned to a single class. The initial weights are often randomly selected and then updated to obtain the desired result. There are many techniques that are used to update the weights for each neuron. Back-propagation algorithm is method widely using to update the weights of the neural network by calculating the gradient of loss function [42].

In general, ANN consists of three layers, one layer for input, one for output and at least one hidden layer between them. Inputs are passed to the hidden layers by a group of neurons in each layer. Where these neurons are connected to each other by weight, which represents the importance of input value, the more valuable neurons from the other will have a greater impact on the next layer of neurons.

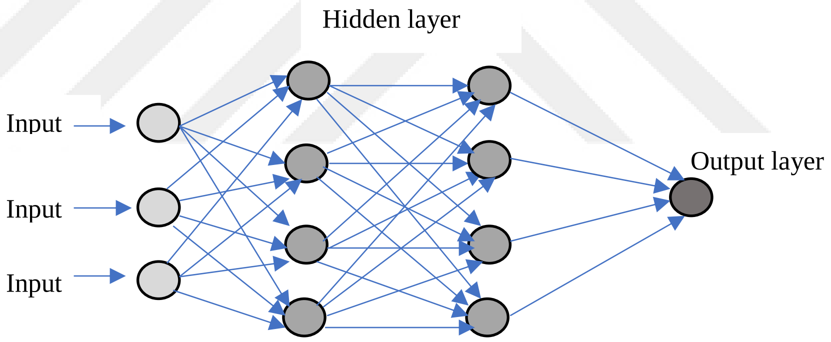

Figure 2.7: Topology of deep neural network

Deep neural networks indicate that there is more than one hidden layer in the neural network, it widely using for supervised and unsupervised. DNN is highly efficient in deriving meanings from complex or imprecise data, extracting patterns and identifying extremely complex directions that cannot be known by humans or other computer technologies. The trained neural network can be considered an "expert" in the category of information given for analysis. This expert can then be used to make projections with new cases of interest. Neural networks are capable of handling large and noisy data, Where the greater the size of the data, the more accurate modeling results [43].

Various types of artificial neural networks (ANN) have been developed, the first and simplest neural networks that are widely used is feedforward neural network , Which proposed in this work as one of the machine learning techniques to be applied, in this type of networks, information is transmitted in parallel from the input layer directly through the hidden layers and then into the output layer without cycles/loops, the proposed neural network contains three hidden layers. each network layer is fully connected to the next layer in the network, where ReLU function using in hidden layers and sigmoid function used in the output layer for binary classification while the SoftMax function used in the output layer for multiclass classification., where Backpropagation for learning the model.

Ilyas Benmessahel, et al. [44] Developed advanced approach used new natural evolutionary algorithm (EA) called multiverse optimizer (MVO) is investigated and combined with an artificial neural network (ANN). This main idea of this approach is to train the feedforward multilayer artificial neural network using MOV and applied on UNSW-NB15 and NLSKDD. The results demonstrate the effectiveness of the proposed approach where the accuracy rate was 99.61% on UNSW-NB15, while the NLSKDD achieved 98.21%.

Sayantan Guha, et al. [45] Presented approach using an artificial neural network with a genetic algorithm to reduce the number of features extracted from the network traffic data. The fully connected feedforward multilayer neural network used to classify network traffic data and consist three-layer input, a hidden layer and output layer. The nodes number in input layer has the same number of feature selection selected by the genetic algorithm. The output layer has a number of nodes as the same number of attack categories. While the ANN includes one hidden layer that consist of 22 nodes. This topology applied on tow datasets UNSW-NB15 and NLSKDD. The results of this approach are 95.46% and 91.98% respectively of datasets.

Khoi khac Nguyen, et al. [46] Authors proposed a novel framework to detect anomaly in mobile cloud environment by using deep learning approach. Where the deep learning model includes two steps: feature analysis and dimension reduction, the aim of feature analysis is extracted features and learn from the features. While the dimension reduction using Principle Component Analysis (PCA) to reduce the dimensionality of dataset. learning process includes three layers, input, output and hidden layers. Each layer act step, the first step: pre-learning process, where the Gaussian Binary Restricted Boltzmann Machine (GRBM) using to transform real value, input data to binary code that use in the hidden layer. Second step: deep learning phase adjust the weights of the neural network. Finally step: SoftMax Regression is the output layer that receives the output from the last hidden layer to classify the packets. three datasets used to evaluate the framework UNSW-NB15, NLSKDD and KDDCUP99. The accuracies rate is 95.84%, 90.99% and 97.11% respectively.

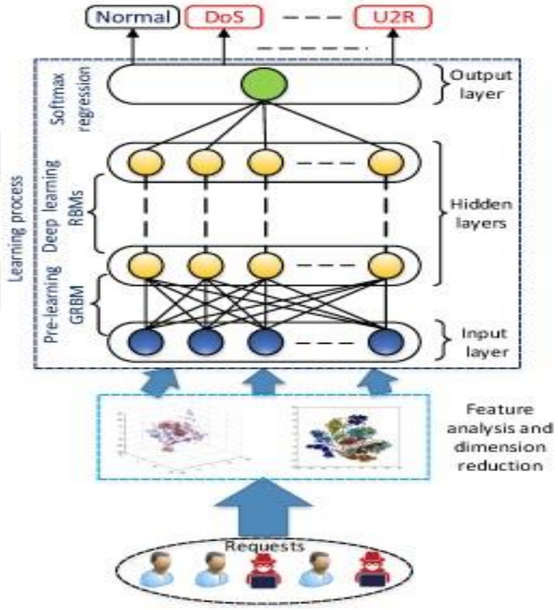

Figure 2.8: DNN with Features Analysis and Dimension Reduction approach [46]

Muna AL-Hawawreh, et al. [47] Proposed deep-learning methods for unsupervised learning. Automatic dimensionality reductions, deep auto-encoder (DAE) and deep feedforward neural network (DFFNN) architecture using to consecutive training process on two dataset UNSW-NB15 and NLSKDD. The network structures of the model were used for both datasets is input layer with 41 nodes, three hidden layers (10,3,10 nodes) and output layer consists 41 nodes for DAE. While 2 nodes of output layer for DFFNN. This method achieved accuracy rates are 98.4% and 92.5% NLSKDD and UNSW-NB15 respectively.

The ensemble is one of the most important methods used in machine learning to minimize noise, bias and variance factors. Which aims to improve the stability and accuracy of machine learning algorithms. The ensemble is a set of predictions that are integrated together to obtain a final prediction where several different predictions are combined to reach the target prediction, ensemble techniques are classified into two methods Boosting and Bagging.

Table 2.1: Ensemble techniques types

Ensemble techniques	**Bagging**	e.g., Random Forest Independent classification Handle overfitting Reduce variance and increases accuracy
	Boosting	e.g., Gradient Boosting Tree Sequential classifiers Reduce bias, variance and increases accuracy Can overfitting

Bagging (Bootstrap Aggregation) depends on the creation many independent predictors (Decision Tree). Which are trained on subsets of data and then collect these predictions using one of the models averaging techniques weighted average, vote majority or normal average. Using a set of decision trees that are trained on subsets of data gives more accurate results than using a single decision tree and reduce the variation of the decision tree. Random forest algorithm is bagging ensemble. Boosting is an ensemble technique, Different from Bagging in that the predictor is not created independently but sequentially. Where every tree that is generated is trying to learn from the errors of the tree before it. Each time the input is incorrectly classified by a hypothesis, weight is increased so that the following hypothesis is more likely to be correctly categorized Gradient Boosted Tree is boosting ensemble [48].

Random forest is a supervised learning algorithm that is one of the most used algorithms because it’s flexible and easy to use for classification and regression tasks. it depends on the ensemble method where it creates a set of decision trees and combines them to obtain a higher accuracy and stability prediction. Random forests rely on the decision tree algorithm to build trees and training of each decision tree that is generated in parallel with a different subset of the training data. The nodes of each decision tree are split using a randomly selected attribute of the data. Randomly selected attribute leading to the distribution of potential errors evenly throughout the model and being eliminated through the majority voting decision strategy in the model.

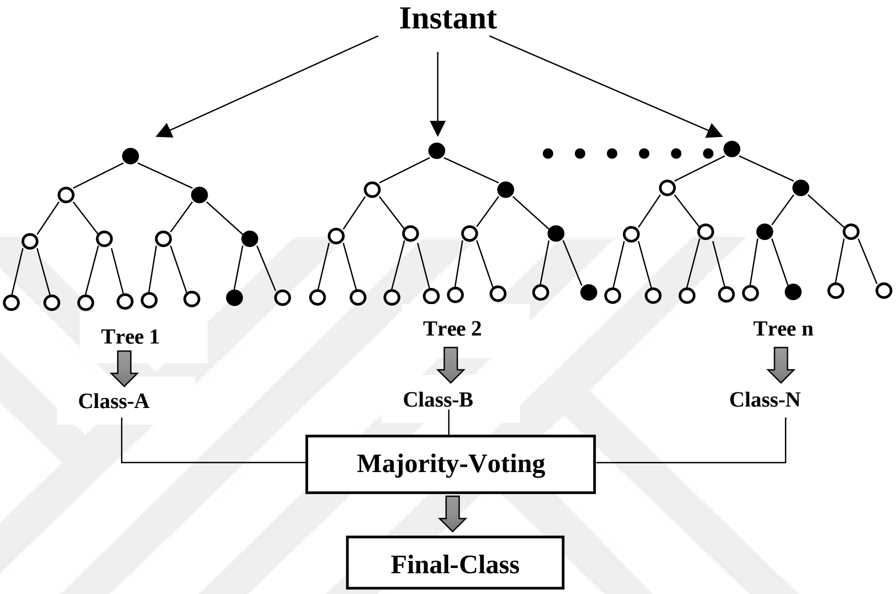

Figure 2.9: Random Forest simplified [49]

A random forest is defined as a collection of tree-structured classifiers {h (x, Θk), k=1,…} where the {Θk} are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x [50].

The growth of a single decision tree can be described within random forests in the following steps: Several random sub-sampling sets are created from the basic data set with replacement. The features are selected with the same sampling approach where a subset of the features is randomly selected from the sum of the overall features. Training the decision tree on each sample. Decision trees grow without pruning. Integrate all the decision Trees predictions results of by simple majority voting. the more trees are built, the more random forests ability to resist noise and increase the classification efficiency. Furthermore, the ability of a random forest algorithm to operate within distributed and parallel computing and efficiently process data by simple classifiers built by the decision tree gives it the advantage of scalability and adaptation to large changes in data volume and variety. The random forest includes a set of parameters that need to be adjusted and which often result in improved performance. The two most important Parameters are the number of trees in the forest and a maximum depth of each tree. Increasing the number of trees helps to improve the testing time accuracy and reduce the variance in predictions, while the increases the depth of trees helps to build a more cohesive and powerful model [51,52].

Because of its high performance in classification, the random forest has been proposed in many approaches and methods to improve and develop intrusion detection systems. Malik, et al. [53] provided IDS based on binary particle swarm optimization (PSO) and Random forest (RF). Where the binary PSO is used to search and select a more suitable set of attributes to classify network interferences and RF is used to classification. The proposed method consists of two steps to feature selection and classification, 10 cross-validation used to evaluate the classifier, all experiments were implemented in MATLAB4.

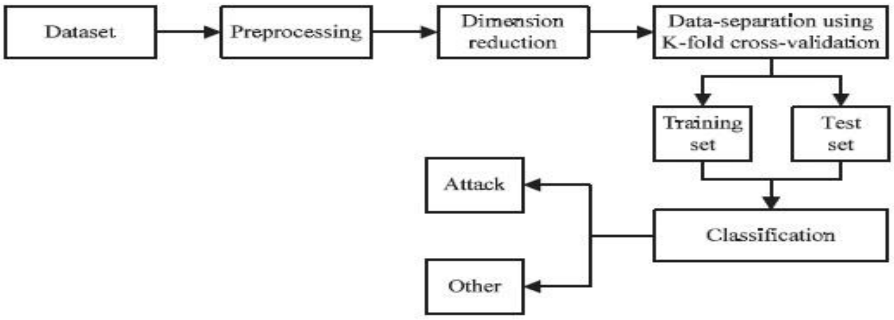

Figure 2.10: Diagram of Random Forest and dimension redection [53]

Random forest molding-based intrusion detection provided by Farnaaz and Jabbar [54] to detect four types of attack included in NLS-KDD dataset like Dos, Probe, U2R and R2L. the proposed method using feature selection techniques (Symmetrical uncertainty (SU)) to select a subset of features from the dataset, reduce dimensionality, remove redundant and irrelevant features. 10 cross-validation using for classification, the experiments implemented on WEKA tool. This work using a set of performance measures to evaluate the classifier like accuracy, Detection rate, False alarm rate and

Mathews correlation coefficient (MCC). The accuracies of each attack are 99.68% for DOS, 99.63% for Probe, 99.69% for R2L and 99.68 for R2L.

Shi et al. [55] This paper discussed the used Semi-Supervised Random Forest for IDS, the classifier applied on KDD 1999 dataset, that included 41 features and four types of attacks. The results compared with the supervised RF and semi-supervised Ladder Network. These classifiers applied on four sets of samples selected from the dataset. Two performance measure used to evaluate classifiers are accuracy and execution time, supervised RF performed better than the semi-supervised RF in accuracy and execution time, while the semi-supervised RF achieved results better than semi-supervised Ladder Network of both measures.

Iman Sharafaldin et al. [56] the authors produce a reliable CICIDS2017 dataset that contains benign and seven common attack network flows. Examine the performance and accuracy of the selected features with seven common machine learning algorithms K-Nearest Neighbors, Random Forest, ID3, Ada-boost, Multilayer Perceptron, Nave Bayes and Quadratic Discriminant Analysis. According to the accuracy and execution time results the RF algorithms achieved best results, the accuracy is 98%. In addition, the new dataset was compared with publicly available data sets from 1998 till 2016 based on 11 criteria representing common errors and criticisms of previous data sets, the comparison results showed that the new dataset addressed all errors and criticisms.

Gradient Boosted is supervised machine learning, widely used for classification and regression tasks. GBT is an ensemble method as RF, But the difference is in the predictor’s creation (Decision trees). GBT is based on weak learners (high bias, low variance), weak learner in decision tree mean shallow tree. Where the GBT start with the shallow tree to build predictor, then calculate the error of expectations and passes the errors to the second tree as a target. The second tree adopts the new prediction model according to the data of the first tree model. The error is calculated for the new predictor model and passed to the third tree and so forth. GBT aims to reduce loss function on the training dataset and using the (log loss) for classification to reduce the loss function [57].

The log loss definition as:

$$2\sum_{i=1}^{N}\log(1+exp(-2y_iF(x_i))) \text{ ....(2.5)}$$

where $\mathrm{N} = \text{number of instances}, y_i\ lable\ of\ instanc\ i, x_i = features\ of\ instance\ i,\ \ f_{(xi)}$= model’s predicted label of instance i [51].

Tama and Rhee [58] Proposed anomaly-based IDS using gradient b**oos**ted machine (GBM), where the best parameters of GBM are obtained by performance grid search. GBM applied on three datasets UNSW-NB15, NLSKDD and GPRS with tenfold cross-validation. The performance of proposed approach compared with four classification techniques, Random forest (RF), deep neural network, support vector machine, classification and regression (CART). Using four measures accuracy, specificity, sensitivity, and AUC metric. The GBN classifier achieved high results on the three datasets.

Gupta and Manish [59] This paper suggests a framework for intrusion detecting based on feature selection using two algorithms, correlation-based feature selection and Chi-squared feature selection. Moreover, five classification algorithms using to detect the intrusion are Logistic regression, Support vector machines, Random forest, Gradient Boosted Decision trees and Naive Bayes. KDD’99 and NLS_KDD datasets are used to evaluate the framework. Five measures to performance evaluate (accuracy, training time, prediction time, specificity and sensitivity), the Random Forest and GBT achieved the best accuracy on both datasets.

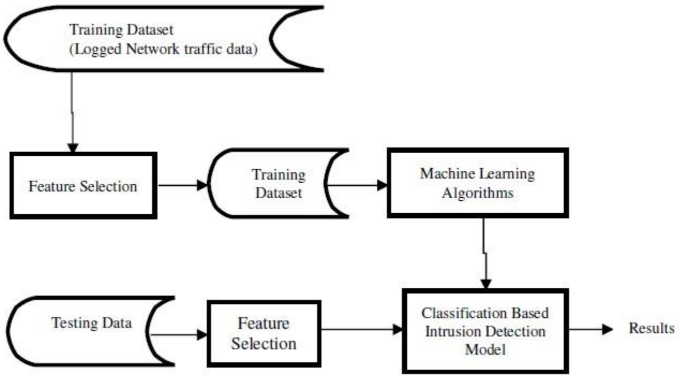
Training Dataset
(Logged Network traffic data)
Feature Selection
Training Dataset
Machine Learning Algorithms
Testing Data
Feature Selection
Classification Based Intrusion Detection Model
Results

Figure 2.11: A framework of feature selection and classification techniques [59]

Cross-validation is also called rotation estimation. It is a common statistical method used to estimate the efficiency of machine learning models. Where This technique is used in applied machine learning to compare and select a case model in predictive modeling. Because it is easily understood and applied, and results in efficiency estimates that often have low bias than other methods. K-fold cross-validation one of the most famous methods used for cross-validation. It can identify and determine the best parameters that provide higher efficiency and accuracy in the model. Where it provides a good assessment of how the model works on the entire data set thus reducing the variance in the data set and takes all the folds into the training account as well as the test and thus reduces the variation in the data set. Which is why it provides a good indication of the generalization error [60].

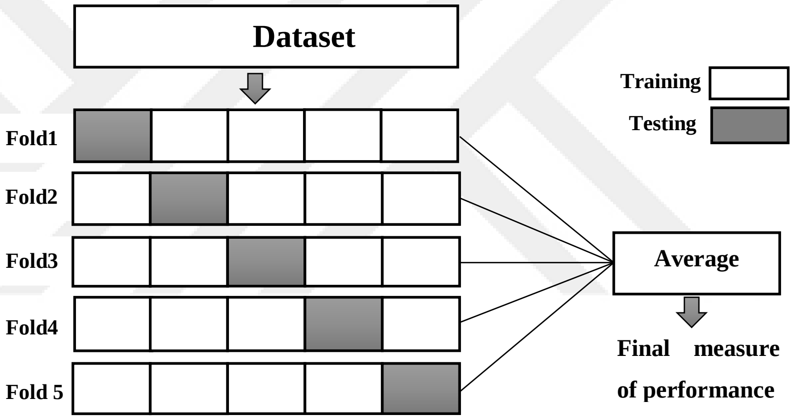

Figure 2.12: 5-fold cross validation data split

K-Fold Cross-validation technique has one parameter called (K) indicates the number of groups to which the data set will be divided, this procedure is implemented in several steps:

Table 2.2: Main steps of 5-cross validation

Step1	The data set is Shuffle randomly
Step 2	Divide data set to equal k sets
Step 3	assume that K = 5 and folds: F1 to F5.
Step 4	For i = 1 to i = 5, one Fi set is selected for the testing and the remainder F2 to F5 for the training.
Step 5	This procedure is performed k times, where each subset is used as a test set once, while all subsets are used as training set several times.
Step 6	Train the machine learning on the subsets of training to create the model and evaluate it on the subset of testing to get the accuracy rate, Where the accuracy rate is kept, and the model is ignored.
Step 7	Finally, Estimate the accuracy of the model by averaging accuracy rate for all results in k cases of cross validation.

MalekAl-Zewairi, et al. [61] A deep learning model was proposed based on artificial neural networks using back-propagation and stochastic gradient descent method. Where the model evaluated as a binomial classifier for NIDSs on UNSW-NB15 datasets. DL model contained five hidden layers in each layer of ten neurons and the 10-folds Cross validation technique was used. The results showed that the model obtained high accuracy and low alarm rate compared to earlier models.

Pritma and Tama [62] proposed compares the performance of IDS by applying a random forest classifier with respect to two performance measures, accuracy and false alarm rate, and used 10fold cross-validation technique. Three datasets of IDSs used in the experiment are NSL-KDD, UNSW-NB15, and GPRS. where Compared the results with the set of classifiers as Multilayer Perceptron, Decision Tree and NBterr. The results of the study show the effectiveness of the proposed model using a cross-validation technique, while the comparison of the results of the proposed model and an MLP classifier. The accuracy of the proposed model was better than the MLP.

Deep learning techniques often require large volumes of data to ideally train and build the model, resulting in a long training time. Therefore, this work suggests the use of big data computing techniques because of the characteristics afforded by the ability to deal with the size of data and very speed in the process of data analysis. Apache Spark is a cluster computing technique designed to handle big data Rapidly. Spark developed to address MapReduce's limitations and more effectively support many types of computation, including streaming processing and interactive queries. Spark has many features, one of the main features is the ability to run computation in memory, where the data is stored and processed in the memory without being transferred to the disk. Which provides high efficiency in the detection of patterns interactively without waiting for minutes and hours. Spark covers a wide range of workloads that require separate techniques to implement [63].

Apache Spark includes a machine learning library that contains a range of machine learning algorithms. That are widely used in many fields. Recently, there has been a great need for deep machine learning techniques that are highly efficient in dealing with large data and provide high performance in extracting patterns. A collection of open source deep learning libraries has been built, such as TensorFlow, Keras, Theano etc. The Spark machine learning Library does not include deep learning techniques but can integrate some of these libraries.

This work suggests using a Keras library, which is high-level neural network API and provide fast and easy way to create a range of deep learning models by python. Where it is run on top of TensorFlow or Theano and can execute on Graphics processing unit (GPU), Central Processing Unit (CPU) or Tensor Processing Unit (TPU). Keras has been developed to act as a distributed, providing a deep learning distributed framework. That executed on top of apache spark and Keras with included modern distributed optimization algorithms to reduce the training time using distributed deep learning techniques. Distributed Keras supported several distributed methods like models using data parallel methods and the training of ensembles.

Recently, several studies have suggested using Apache Spark to enhance the performance of intrusion detection systems. P. Dahiya, et al. [64] proposed an intrusion detection system using Apache Spark to implement the proposed approach which relies on the feature reduction algorithm, Canonical Correlation Analysis (CCA) and Linear Discriminant Analysis (LDA) and seven well-known classification algorithms Nave Bayes, REP Tree, Random Tree, Random Forest, Random Committee, Bagging and Randomizable. The results show that the approach is more efficient and fast using (LDA) and Random Tree algorithms, and achieved the best accuracy better than other algorithms.

Mustapha Beluch, et al. [65] Authors evaluated the performance set of classification algorithms (SVM, Decision Tree, Naive Bayes, Random Forest) using Apache Spark and complete the UNSW-NB15 dataset with all 42 features. accuracy, sensitivity, specificity and execution time are used to evaluate the performance of classifiers. The results showing the Random forest classifier performed better than other classifiers in four measures, where its accuracy is 97.49%.

CHAPTER 3

METHODOLOGY

The proposed method includes a set of steps that begin with the preprocessing of the datasets and then selection of related features. Where the K-means clustering (homogeneity metric) is used as an unsupervised feature selection technique for the selection of relevant features from two datasets to improve the performance of classifiers. Five-fold cross validation is used to estimate and improve the performance of machine learning models. Deep neural network and two ensemble techniques (RF, GBT) are used to extract the models from the subsets of relevant features. The proposed method is evaluated on two recent datasets, namely UNSW-NB15 and CICIDS2017, which contain a combination of common and modern attacks. The datasets are preprocessed to be suitable for applying the machine learning techniques.

3.1. Dataset Preprocessing

To provide a more suitable data for the neural network classifier and ensemble techniques, the dataset is passed through a group of preprocessing operations. These operations are summarized below:

- Remove socket information: As the original dataset includes the IP address and Port numbers of the source and destination hosts in the network. It is important to remove such information to provide unbiased detection, where using such information may results in overfitted training toward this socket information. However, it is more important to let the classifier learn from the characteristics of the packet itself, so that, any host with similar packet information is filtered out regardless to its socket information.

- Remove white spaces: Some of the multi-class labels in the dataset include white spaces. Such white spaces result in different classes as the actual value is different from the labels of other tuples in the same class.
- Label encoding: The multi-class labels in the dataset are provided with the attack's names, which are strings values. Thus, it is important to encode these values into numerical values, so that, the classifier can learn the class number that each tuple belongs to. This operation is executed using the multi-class labels only, as the binary labels are already in zero-one formation.
- Data normalization: The numerical data in the dataset are of different ranges, which poses some challenges to the classifier during training to compensate these differences. Thus, it is important to normalize the values in each attribute, so that, the minimum value in each attribute is zero, while the maximum is one. This provides more homogeneous values to the classifier while maintaining the relativity among the values of each attribute.
- Remove/replace massing and infinity values: CICIDS2017 dataset contains 2,867 tuples as missing and infinity values, this has been addressed in two ways that s produces two datasets: the first, the dataset is without the missing and infinite values, where is removed all missing and infinity values. the second dataset is replaced the infinite values with the maximum value and the missing values with the average values. Both datasets are used to evaluate the proposed method.
- For multiclass classification, Information packets that represent normal network traffic from both data sets are ignored and only the attack information packets are using to evaluate the proposed method.

3.2. Feature Ranking (homogeneity metric)

After the preprocessing phase, as showing in Algorithm 3.1 the K-means clustering algorithm is applied to the two datasets for features ranking. The technique is using to do the features ranking, for features selections, is that taking each attribute separately, then use it to cluster the dataset. In binary classification K= 2, that mean the data point of feature clustering to two groups, normal or anomaly. For multi-class classification the K equals the number of attacks in datasets. Thereafter the homogeneity score is calculated of the resulting clusters are then used as a rank for that features. Such score indicates that better classification can be conducted relying on that feature, while a lower score indicates that this feature does not have a significant role in the classification.

When the rank of homogeneity is determined for each feature, they are arranged in descending order from the lowest rank to the highest rank. The homogeneity rank is between zero and one. The zero refers to the lack of homogeneity between the data points of the feature in the cluster. Therefore, this feature belongs to different classes. It indicates high homogeneity between feature data points and this means that this feature represents only one class. Finding a unique feature that belongs to a single class is probably more effective than relying on common features in the classification process, especially with the increasing volume of heterogeneous data generated from several sources.

Algorithm 3.1: Steps of the proposed methods

```
Inputs: C=number of classes; F=number of features; R=Features ranks D=Data;
L=Labels
Output: Selected features

Step1: hs=empty array(F,2)   //initiate an empty array to store that homogeneity
score of each attribute.
        Acc=0   //Declare a variable to calculate accuracy.
        Tacc=empty array(F,2)   //initiate an empty array to store accuracies.

Step2: for f=1 to F
          d=D[:,f]   //Select single attribute from the data.
          cl=Kmeans(d)   //Cluster the data using the selected attribute's values.
          hs[f]=[f,cl]   //Insert the feature number and homogeneity score.

Step3: or=order(hs,1,ascending). //Order features in ascending order based on their
ranks.

Step4: for f=0 to F
          sf=rd[f:,1]   //Selected higher ranked features.
          d=D[:,sf] //Filter out lower ranked features.
          For train, test in Kfolds(d,5)
                Trclsf=clsf.train(d[train], L[train]) //Train a classifier using the
training                set
                prd=Trclsf.predict(d[test])   //Predict labels for data instances in the
test                    set.
                Acc=Acc+accuracy_score(L[test],prd)
          Tacc[f]=[f,Acc/5)   //Store the average accuracy of five folds.

Step5: bfi=Tacc[argmax(Tacc[:,1]),0]   //Find the index of highest accuracy.
       bfl=or[bfi:]   //Retrieve the list of features that achieved heist accuracy.
       Return(bfl)
```

3.3. Implemented Artificial Neural Network and Ensemble Techniques

As in Figure 3.1, the deep neural network, Random Forest and Gradient Boosted Tree Classification algorithms were implemented on CICIDS2018 and UNSW-NB15 datasets to evaluate the proposed methods. Two scenarios are executed using amazon web server EC2 (AWS). Each algorithm was applied across two scenarios binary classification and multiclass classification except GBT, because spark MLlib does not support it for multiclass classification.

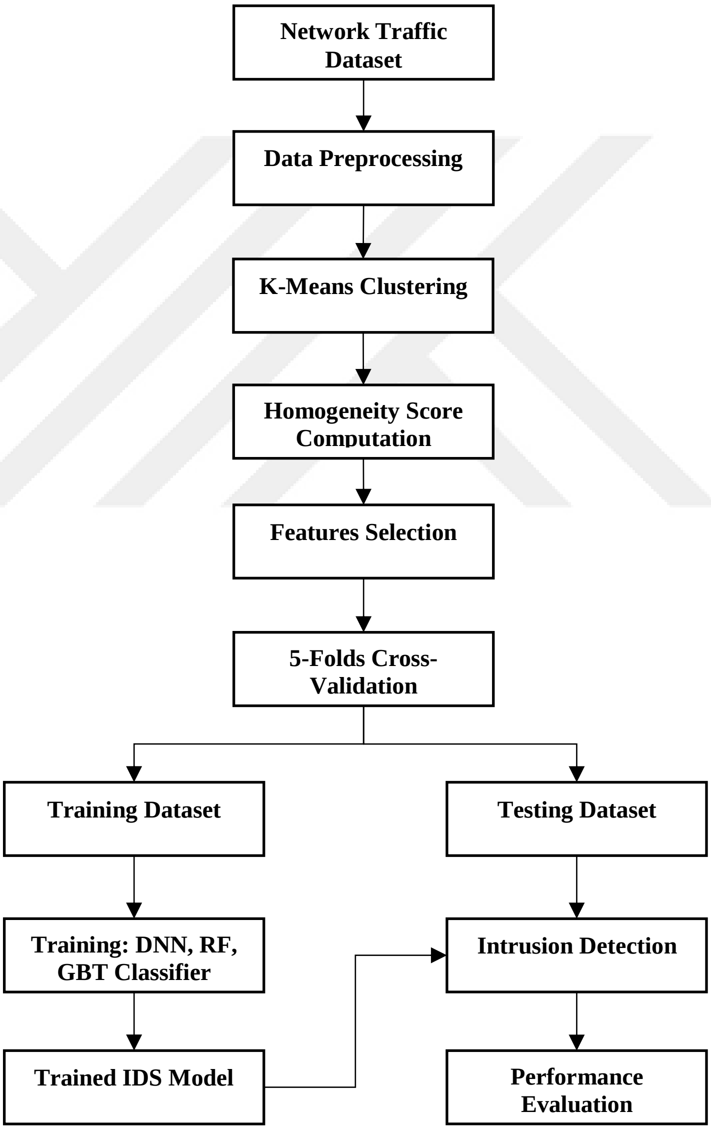

Figure 3.1: Flowchart of proposed method

Deep neural network considers one of feedforward artificial neural networks, DNN consists of Multiple layers of nodes. Each layer is fully connected to the next layer in the network. Where 43 and 78 nodes in input layer that represent the number of features in UNSW-NB15 and CICIDS2017 dataset Respectively. Three hidden layers 128,64 and 32 nodes per layer respectively, ReLU activation function used in the hidden layer. SoftMax function used in the output layer for multiclass classification and sigmoid function for binary classification. Where Backpropagation for learning the model, training epoch 1000 (Epoch means one pass over the full training set) and Batch size 1,000,000 (Total number of training examples present in a single batch).

3.4. Binary Classification

The first scenario uses binary classification that refers to the result of the classification process being two groups. Predict each packet to be a part of a normal or attack traffic. Machine Learning techniques are initially applied to full two datasets, then remove the first feature from the dataset and repeat Calculates the evaluation metric. Repeated removing one feature each time of the data set and calculation the evaluation metric until access to the last feature of the dataset. The deep neural network has the same settings as above except the output layer which has only two nodes and sigmoid function is used. The classifiers are applied on both dataset, UNSW-NB15 and two sets of CICIDS2017. Before performing assembly techniques, values for parameters are determined. For random forest, there are two important parameters whose values must be determined, the number of trees in the forest that has been given value 100, while the depth of the tree is 4. Gradient Boosted Tree has two parameters, log loss for classification to reduce the loss function and the number of iterations is 10.

3.5. Multiclass Classification

The second scenario is multiclass classification, where classifies the attack packets onto the number of classes in datasets, nine different attacks in the UNSW-NB15 and fourteen in the CICIDS2017. The same settings used for deep learning in binary classifications except the number of nodes in the output layer and the activation function. The number of nodes in the output layer equal to the number of dataset attacks classes and the SoftMax as the activation function. The same settings of random forest. In this scenario, the packets information of attack only using to evaluate the proposed method. To obtain high performance and reduce the bias of machine learning techniques, k-fold cross-validation was used in this work to evaluate the machine learning model. the entire dataset is split into five bins randomly, each bin is used once for testing, while the remaining bins are used for training, per each iteration. Thus, when the evaluation is complete, a prediction per each tuple in the dataset is provided by the deep neural network, Random forest, and the accuracy calculated.

3.6. Description of datasets

There are several data sets created to evaluate the proposed techniques to develop and improve the performance of intrusion detection systems. According to most studies based on intrusion detection systems used the KDD cup 99 dataset that launched in 1999 [66, 67] and the refined version NLS-KDD dataset [68, 69]. Which include four types of attacks: DOS/DDOS, Probing, U2R and R2L. This data sets are relatively old and cannot be relied upon to evaluate intrusion detection systems because they do not contain new types of attacks and modern normal behaviors, especially with the great development in attack methods and the emergence of new types [70]. This section will discuss two modern intrusion detection datasets UNSW-NB15 and CICIDS2017 Which includes a wide range of modern attacks that allow for more realistic evaluation of the proposed approach.

3.6.1. UNSW-NB15 Dataset

The UNSW-NB15 [71,72] is one of the latest datasets created by the cyber security research group at Australian center of cybersecurity (ACCS) for evaluating IDSs. It has become available to researchers since late 2015. Where the IXIA PerfectStorm testing platform [73] is used to generate about 100 GB of normal and abnormal network traffic. Extracted 49 features from the raw “pcap” file by using the Argus [74] and Bro-IDS [75] split into five sets Basic features: Flow features, Content features, Time Features, Additional generated features. The dataset continues a total number of (2,540,047 rec).

As in Figure 3.2, the data set contains nine types of recent and common attacks, namely, Fuzzers, Analysis, Backdoors, Dos, Exploits, Generic, Reconnaissance, Shellcode and worms. A total of its records in the data set (321,283 rec), while the total number of normal records is (2,218,764 rec). The size of the normal information packets represents 88% of the data set size, while the attack information packets represent 12%. The UNSW-NB15 dataset has many advantages that make it one of the most important data sets that have been widely used in evaluating intrusion detection systems. The information packets of UNSW-NB15 dataset collected over 16 hours and 15 hours that represents modern traffic and common attack scenarios, also, it Contains a set of features that represent payload and header for packet data making it reflect network packets efficiently.

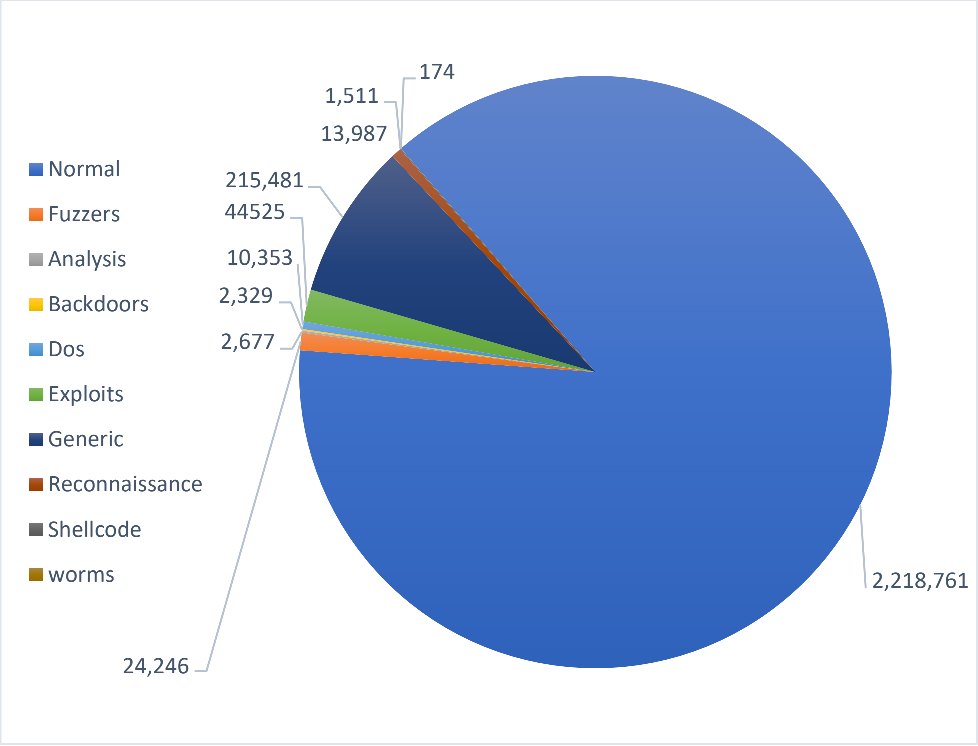

Figure 3.2: Class distribution in UNSW-NB15 dataset

Table 3.1 shows the types of attacks contained in the UNSW-NB15 dataset and the number of their records, as well as the description of each attack:

Table 3.1: Description of classes in UNSW-NB15 dataset

No.	Category	No. of Records	Description
1	Normal	2,218,761	Natural transection data
2	Fuzzers	24,246	It tries to enter a set of random data for the program and analyzing the results to find errors may be exploitable
3	Analysis	2,677	It is the process of intercepting and checking network traffic in order to collect information from patterns in communication It contains different attacks of PortScan, spam and html files penetrations
4	Backdoors	2,329	It is a technique designed to bypass the protection system without detection to access the computer or network resources.
5	Dos	10,353	It is an attempt to make a server or network resource unavailable by interrupting or suspending host services
6	Exploits	44525	An attacker is trying to get a security vulnerability in an operating system or in a program and exploit it to access network
7	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.
8	Reconnaissance	13,987	Includes a set of attacks aimed at collecting information about the network and its sources
9	Shellcode	1,511	It is a piece code that is carefully loaded with an application and executed as soon as the application runs to control the hacked device and access network resources.
10	worms	174	Is a self-replicating malware that relies on duplicates itself on the computers and is intended to consume system resources which slow down or stop some services

3.6.2. CICIDS2017 Dataset

The CICIDS2017 dataset [55,76] released in late 2017 via Canadian Institute for Cybersecurity (CIC), where it contains benign and the most up-to-data common attacks. B-Profile system [77] used for the abstract behavior of human interactions and to generate benign natural background traffic and CICFlowMeter [78] to extract the features from the dataset. CICIDS2017 Dataset contains the most common attack based on the 2016 McAfee report (Dos, DDos, Web-based, Brute force, Infiltration, Heart-bleed, Bot and Scan) with more than 80 features extracted from the generated network traffic. For created the dataset complete network topology used where includes different operating systems (Windows, Ubuntu, MAC OS X) and network devices (Modem, Switches, Firewall, Routers).

Figure 3.3 shows the distribution of each attack in the CICIDS2017 dataset. it Contains 14 types of attacks and (2,273,097 rec) of normal packets information (BENIGN), Which represents 80% of dataset size. While (557,646 rec) of attack packets information that represents 20% of dataset size. The dataset provides a variety of common and recent attacks, that is useful for a more efficient and realistic assessment.

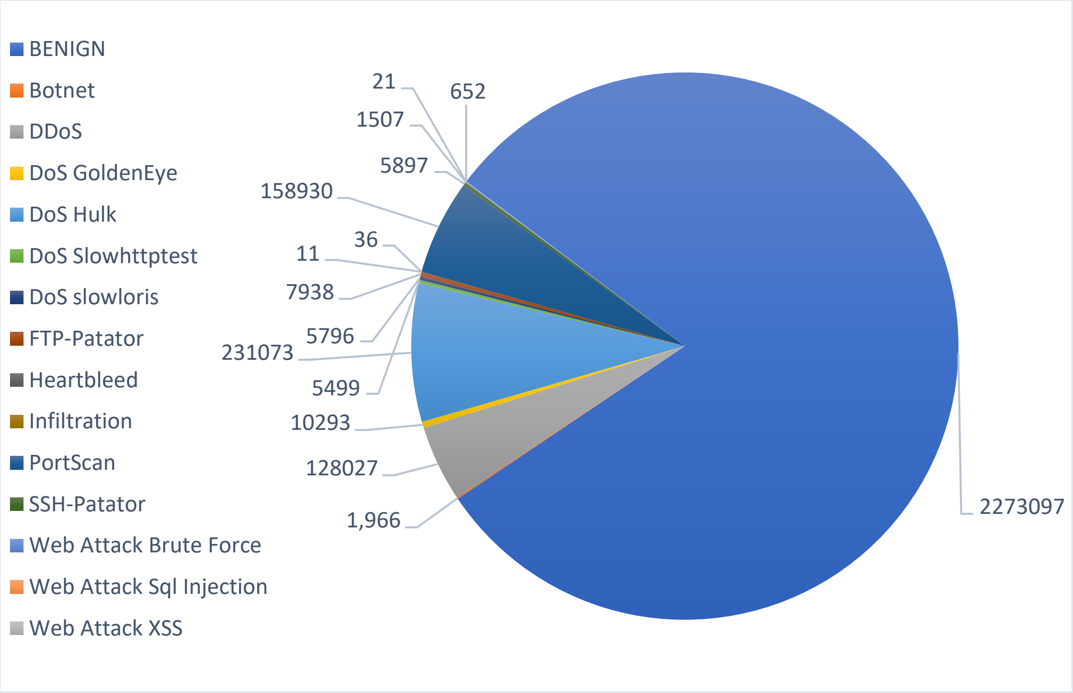

Figure 3.3: The classes distribution of CICIDS2017 dataset

Table 3.2 shows a brief description of the 14 types of attacks contained in the data set and the number of records for each attack.

Table 3.2: Description of the classes in CICIDS2017 dataset

No.	Category	No. of records	Description
1	BENIGN	2273097	Normal human activities
2	DoS Hulk	231073	Generate a huge number of unique and obfuscated network traffic in the web server to bypass the caching engines then obstruction or stop web server resources.
3	PortScan	158930	This attack is done by using malicious techniques to scan computer ports and exploit detected vulnerabilities to access network or server resources
4	DDoS	128027	A distributed denial-of-service (DDoS) attack Is to try to disrupt the normal traffic of the target server or network by flooding the target with a flood of traffic exploiting a group of hacked computers as a source of attack traffic.
5	DoS GoldenEye	10293	It aims to consume all available sockets on the HTTP server using KeepAlive paired with cache-control option to persist socket connection busting through caching
6	FTP-Patator	7938	It is a type of brute force attacks, that try to crack passwords using two method dictionary attack or generate password.
7	SSH-Patator	5897	It Is a type of brute force attacks that try to crack passwords using two method dictionary attack or generate password.
8	DoS slowloris	5796	Works by sending several incomplete requests to connect to the server and keep these connections open for as long as possible, which leads to the exploitation of the maximum communication and thus denial normal communications.

9	DoS Slowhttptest	5499	This attack is intended to drain the memory and CPU of the server by sending incomplete requests to the HTTP server so that the server keeps these requests until completion, which later leads to denial of server.
10	Botnet	1966	A set of networked computers that are controlled by sending malicious software to stop or take down the network service, web site, steal dated, access to the network resource.
11	Web Attack Brute Force	1507	It uses a set of consecutive automated programs to create a large number of guesses to get passwords or PIN
12	Web Attack XSS	652	Cross-Site Scripting attacks Refers to the attacker injecting malicious code into a Web site or Web application to access the victim's machine and thus directing the victim to malicious sites or access to his private data.
13	Infiltration	36	The first step in cyber kill chain attack that attempting to perform a data breach, it is Exploiting security vulnerabilities in applications or sending malicious files to access network resources.
14	Web Attack Sql Injection	21	Refers to malicious SQL code that Injection in web application for database manipulation to access secret information.
15	Heartbleed	11	Attack targets a serious vulnerability in the popular OpenSSL cryptographic software library, where allows an attacker to retrieve a block of memory of the server up to 64kb in response directly from the vulnerable server via sending the malicious heartbeat.

CHAPTER 4

EXPERIMENTAL RESULTS

This section represents the results and evaluated of two scenarios, binary and multiclass classification. The experiments are conducted using Amazon’s Elastic Map Reduce (EMR) cloud services with PySpark setup. A cluster of ten nodes is used for these experiments, where each node has a 2.4 GHz Intel Xeon® E5-2676 v3 processor and 64GB of memory. The random forest and GBT classifiers are implemented using Spark’s built-in machine learning library MLlib. The deep learning model is implemented using the Distributed Keras library, which implements and operates Keras deep learning models using a PySpark cluster. The performance of deep neural network and ensemble techniques are evaluated in terms of accuracy, prediction time and confusion matrix, which are calculated as follows:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.1)$$

Confusion matrix:

Table 4.1: Confusion Matrix

Actual Class	Predicted Class	
	Attack	Normal
Attack	TP	FN
Normal	FP	TN

The true positive (TP) is the attack instances correctly classified as belonging to the attack class. False positive (FP) is the normal instances incorrectly classified as belonging to the attack class. True negative (TN) is the normal instances correctly classified as belonging to the normal class. False negative (FN) is the attack instances incorrectly classified as belonging to the normal class [76]. The evaluation of the proposed method are done by two scenarios, the first is binary classification and the second is multiclass classification, where the all classifiers experiments are implemented on two datasets UNSW-NB15 and CICIDS2017 in two scenarios.

4.1. Binary Classification

4.1.1. UNSW-NB15 Dataset

The proposed deep machine learning and ensemble techniques have been applied to a UNSW-NB15 dataset independently. using cross-validation performance evaluation, the entire dataset is split into five bins, each bin is used once for testing, while the remaining bins are used for training, per each iteration. Thus, when the evaluation is complete, a prediction per each tuple in the dataset is provided by the deep neural network and ensemble techniques, and the accuracy, prediction time and confusion matrix computed. The accuracy is computed of full dataset and after deleted the features one by one, then the best accuracy is achieved by the subset of features are selected.

The topology of the deep neural network used in all experiment in binary classification contains five layers, one input layer with 15 nodes, three hidden layers with 128,64 and 32 layers respectively and an output layer with 1 neuron. The average time on full dataset taken by the binary classifier to predict the state of a packet to be a normal or an attack packet is 1.43 ns and accuracy rate is 99. 16%. The best accuracy achieved with the subset of features contain 41 features, which is 99.19% and the predicted time is 1.35 ns. The confusion matrix shown in Table 4.3 illustrates the actual and predicted classes of the tuples in the dataset.

Table 4.2: Result of binary classification DNN with UNSW-NB15 dataset

	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99. 16	1.43	99. 19	1.35	41

Table 4.3: Confusion matrix of DNN binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	287833	33450
Normal	19466	2199298

A random forest classifier with 100 trees, is used in this experiment. The average prediction time on per each tuple in the UNSW-NB15 dataset, to provide a binary classification is 12.13 ns, and the accuracy rate is 98. 85%. the predictions provided by the classifier are summarized in the confusion matrix shown in Table 4.5.

Table 4.4: Result of binary classification RF with UNSW-NB15 dataset

	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
RF	98. 85%	12.13	98. 86%	11.36073	36

Table 4.5: Confusion matrix of RF binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	294310	26973
Normal	2087	2216677

In this experiment, the gradient boosting tree classifier using to classify tuples into two classes, anomaly or normal. Table 4.6 shows the accuracy rate when the classifier is implemented on full UNSW-NB15 dataset, that is 97.83 %. While the average time of predict is 0.70 ns. A subset of features is consisting 26 features achieved best accuracy rate is 97.92% and the average time of predict is 0.37.

Table 4.6: Result of binary classification GPT with UNSW-NB15 dataset

	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
GBT	97.83	0.7004	97.92%	0.37	26

Table 4.7: Confusion matrix of GPT binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	287833	33450
Normal	19466	2199298

Table 4.8 includes the features ranks calculated by the homogeneity metric and arranged in descending order in a matrix starting from features with lower homogeneity down to features with higher homogeneity. The table also shows the accuracy rate and prediction time for each classifier where a feature is removed each time from the dataset and compute the accuracy of the remaining features set until the last feature is reached.

Table 4.8: Results of feature ranking and binary classification on UNSW-NB15

No. Features	Remove Feature	F- Rank	DNN-ACC	RF-ACC	GBT-ACC
43			0.991675	0.9885	0.9783
42	3	0.00000	0.991898	0.9886	0.9784
41	24	0.00113	0.991903	0.9885	0.9784
40	30	0.00602	0.991466	0.9886	0.9783
39	31	0.00622	0.991416	0.9886	0.9783
38	4	0.00712	0.991899	0.9885	0.9783
37	8	0.00911	0.991851	0.9886	0.9783
36	23	0.00911	0.991544	0.9885	0.9783
35	28	0.00932	0.991434	0.9885	0.9783
34	34	0.01196	0.991404	0.9885	0.9783
33	32	0.01220	0.991377	0.9885	0.9783
32	27	0.02752	0.99169	0.9886	0.9783
31	11	0.02803	0.991562	0.9885	0.9783
30	35	0.03109	0.991196	0.9885	0.9790
29	36	0.03231	0.991627	0.9885	0.9783
28	19	0.04128	0.991339	0.9884	0.9787
27	21	0.04619	0.991492	0.9884	0.9785
26	22	0.04844	0.991355	0.9884	0.9792
25	13	0.09075	0.991169	0.9884	0.9788
24	1	0.09864	0.991322	0.9884	0.9792
23	5	0.11000	0.99138	0.9884	0.9782
22	14	0.15909	0.991193	0.9884	0.9786
21	9	0.17254	0.991334	0.9884	0.9787
20	10	0.17426	0.990726	0.9884	0.9781
19	29	0.17544	0.990726	0.9884	0.9781
18	12	0.18851	0.990866	0.9884	0.9781
17	17	0.18927	0.990612	0.9884	0.9782
16	18	0.18929	0.99086	0.9884	0.9781
15	7	0.19401	0.990418	0.9884	0.9781
14	2	0.19507	0.990551	0.9884	0.9780
13	20	0.19982	0.989806	0.9883	0.9778
12	25	0.20008	0.989716	0.9881	0.9778
11	26	0.22955	0.989766	0.9816	0.9780
10	16	0.23613	0.989693	0.9816	0.9780
9	15	0.29547	0.989474	0.9815	0.9763
8	39	0.29830	0.989497	0.9810	0.9762
7	40	0.31214	0.989376	0.9805	0.9763
6	41	0.37118	0.989578	0.9798	0.9762
5	37	0.38086	0.989227	0.9789	0.9777
4	38	0.40017	0.988241	0.9778	0.9760
3	42	0.40521	0.98782	0.9771	0.9767
2	43	0.40639	0.986622	0.9754	0.9739

4.1.2. CICIDS2017 Dataset

In those experiments, the three algorithms were applied for binary classification on the CICIDS2017data set in two ways: first, to replace the infinite values with the maximum value and the missing values with the average values (Rep-CICIDS2017). The second to remove the missing and infinite value (Rem- CICIDS2017). Deep neural network using with the same settings in previews UNSW-NB15 experiments. Table 4.9 shows the prediction time and accuracy rate on full Rep-CICIDS2017 dataset and subset of features. Table 4.10 shows the confusion matrix of the predictions provided by the classifier used in this experiment.

Table 4.9: Result of DNN binary classification with Rep-CICIDS2017 dataset

Rep-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.72	0.0538	97.73	0.0544	59

Table 4.10: Confusion matrix of DNN binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	520731	36915
Normal	27290	2245807

Deep neural network applied on Rem-CICIDS2017 and the table shows the DNN achieved accuracy rate on the full dataset is 97.71%, while the average prediction time of the classifier in this experiment is 0.0531 ns per each tuple. As in Table 4.11, the subset of 59 features achieved a 97.7% accuracy rate and prediction time is 0.05 ns. Table 4.12 shows the confusion matrix of the predictions provided by the classifier used in this experiment.

Table 4.11: Result of DNN binary classification with Rem-CICIDS2017 dataset

Rem-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.71%	0.053	97.72%	0.052	59

Table 4.12: Confusion matrix of DNN binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	519105	37451
Normal	27017	2244303

Random forest classifier applied in this experiment on Rep-CICIDS2017. The average time consumed by the classifier to provide a prediction for a single tuple is 7.71 ns and accuracy rate is 92.54%. The best accuracy achieved by a subset of features is 92.72% with prediction time is 6.78 ns. The predictions provided by the classifier are compared to the actual labels of these tuples in the confusion matrix shown in Table 4.14.

Table 4.13: Result of RF binary classification with Rep-CICIDS2017 dataset

Rep-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
RF	92.54%	7.71	92.72%	6.78	6

Table 4.14: Confusion matrix of RF binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	359826	197820
Normal	8256	2264841

The Rem-CICIDS2017 dataset used to evaluate the random forest classifier with the feature selection approach, using all features in the dataset the classifier achieved

92.54% accuracy rate. The subset of feature consists of 8 features achieved an accuracy rate of 92.71%. Classification results are summarized in the confusion matrix shown in Table 4.16.

Table 4.15: Result of RF binary classification with Rem-CICIDS2017 dataset

Rem-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time(ns)	Number of Features
RF	92.54%	7.68	92. 71%	6.81	9

Table 4.16: Confusion matrix of RF binary classification of feature selection

Actual Class	Predicted Class	
	Attack	**Normal**
Attack	358467	198089
Normal	7976	2263344

In this experiment, gradient boosting tree classifier implemented on Rep-CICIDS2017 dataset to compute the accuracy and prediction time. A subset of 23 features achieved an accuracy rate of 99.97% with prediction time of 0.53 ns. An average of 1.23 ns is consumed by the classifier to predict the time for each tuple. the performance evaluation measures are shown in Table 4.17. Classification results are summarized in the confusion matrix shown in Table 4.18.

Table 4.17: Result of GBT binary classification with Rep-CICIDS2017 dataset

Rep-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
GBT	0.9981	1.23	99.97%	0.53	23

Table 4.18: Confusion matrix of GBT binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	557340	306
Normal	530	2272567

By using Rem-CICIDS2017, the average time taken by the binary classifier to predict the state of a packet to be a normal or an attack packet is 1.24 ns and the accuracy is 99.81%. While the 21 features are selection achieved an accuracy of 99.99% with average prediction time compute for each tuple is 0.49 ns. Table 4.19 includes the performance summary of the classifier used in this experiment. The predictions provided by the classifier are summarized in the confusion matrix shown in Table 4.20.

Table 4.19: Result of GBT binary classification with Rem-CICIDS2017 dataset

Rem-CICIDS2017					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
GBT	99.81%	1.24	99.99%	0.49	21

Table 4.20: Confusion matrix of GBT binary classification of feature selection

Actual Class	Predicted Class	
	Attack	Normal
Attack	556473	83
Normal	213	2271107

By using the Rep-CICIDS2017 dataset to evaluate the proposed method, the results of features rank, accuracy rate of each iteration shown in Table 4.21, while the Table 4.22 shows the results of proposed method applied on Rem-CICIDS2017.

Table 4.21: Results of feature ranking and binary classification with Rep-CICIDS2017

No. Feature	Remove Feature	F-Rank	DNN-ACC	RF-ACC	GBT-ACC
78			0.9772	0.9254	0.9981
77	3	0.00000	0.9747	0.9256	0.9985
76	4	0.00000	0.9751	0.9255	0.9984
75	5	0.00000	0.9746	0.9254	0.9983
74	6	0.00000	0.9756	0.9257	0.9987
73	32	0.00000	0.9740	0.9256	0.9982
72	34	0.00000	0.9737	0.9255	0.9986
71	35	0.00000	0.9744	0.9258	0.9982
70	36	0.00000	0.9764	0.9253	0.9986
69	38	0.00000	0.9756	0.9254	0.9987
68	56	0.00000	0.9770	0.9254	0.9945
67	57	0.00000	0.9758	0.9255	0.9985
66	58	0.00000	0.9768	0.9255	0.9986
65	59	0.00000	0.9746	0.9254	0.9986
64	60	0.00000	0.9736	0.9256	0.9983
63	61	0.00000	0.9748	0.9260	0.9986
62	62	0.00000	0.9742	0.9252	0.9983
61	63	0.00000	0.9769	0.9253	0.9984
60	64	0.00000	0.9743	0.9255	0.9985
59	65	0.00000	0.9773	0.9255	0.9985
58	66	0.00000	0.9723	0.9258	0.9986
57	69	0.00000	0.9717	0.9257	0.9988
56	70	0.00000	0.9715	0.9256	0.9982
55	72	0.00003	0.9722	0.9257	0.9989
54	73	0.00003	0.9722	0.9258	0.9985
53	33	0.00004	0.9733	0.9258	0.9986
52	50	0.00004	0.9722	0.9258	0.9985
51	20	0.00005	0.9701	0.9255	0.9991
50	15	0.00008	0.9714	0.9259	0.9986
49	46	0.00010	0.9720	0.9256	0.9984
48	51	0.00010	0.9704	0.9257	0.9986
47	26	0.00021	0.9723	0.9256	0.9988
46	8	0.00024	0.9719	0.9256	0.9985
45	17	0.00029	0.9723	0.9254	0.9987
44	27	0.00074	0.9706	0.9257	0.9985
43	37	0.00076	0.9719	0.9258	0.9985
42	25	0.00107	0.9715	0.9256	0.9654
41	22	0.00113	0.9722	0.9255	0.9990
40	16	0.00118	0.9744	0.9258	0.9654
39	74	0.00140	0.9751	0.9259	0.9657

38	71	0.00169	0.9739	0.9248	0.9989
37	30	0.00258	0.9756	0.9253	0.9993
36	10	0.00339	0.9729	0.9255	0.9993
35	7	0.00379	0.9738	0.9254	0.9993
34	9	0.00379	0.9750	0.9257	0.9971
33	54	0.00379	0.9743	0.9256	0.9994
32	52	0.00553	0.9630	0.9252	0.9949
31	67	0.00618	0.9589	0.9254	0.9975
30	76	0.00725	0.9590	0.9254	0.9949
29	29	0.00751	0.9593	0.9250	0.9989
28	31	0.00868	0.9590	0.9252	0.9996
27	45	0.00868	0.9599	0.9250	0.9995
26	68	0.01156	0.9595	0.9250	0.9958
25	48	0.01526	0.9616	0.9251	0.9997
24	28	0.01989	0.9574	0.9251	0.9997
23	44	0.02762	0.9584	0.9249	0.999705
22	47	0.02977	0.9355	0.9252	0.9997
21	49	0.03151	0.9055	0.9252	0.999705
20	1	0.03476	0.9032	0.9253	0.9987
19	2	0.04516	0.9020	0.9252	0.9987
18	21	0.04586	0.9003	0.9259	0.9987
17	78	0.09369	0.9005	0.9261	0.9987
16	19	0.09873	0.9031	0.9263	0.9987
15	24	0.09918	0.9012	0.9264	0.9989
14	75	0.09935	0.8994	0.9263	0.9989
13	77	0.10227	0.8947	0.9267	0.9990
12	18	0.11188	0.8929	0.9267	0.9990
11	43	0.11679	0.8912	0.9269	0.9990
10	23	0.13229	0.8883	0.9270	0.9991
9	12	0.13819	0.8884	0.9270	0.9992
8	41	0.18037	0.8883	0.9271	0.9992
7	53	0.18334	0.8882	0.9272	0.9991
6	39	0.19780	0.8882	0.9272	0.9991
5	13	0.21123	0.8882	0.9271	0.9990
4	55	0.21123	0.8880	0.9258	0.9990
3	40	0.21340	0.8882	0.9259	0.9985
2	42	0.23100	0.8873	0.9024	0.9993

Table 4.22: Results of feature ranking and binary classification with Rem-CICIDS2017

No. Feature	Removed Feature	F-Rank	DNN-ACC	RF-ACC	GBT-ACC
78			0.9771	0.9254	0.9981
77	3	0.00000	0.9747	0.9258	0.9985
76	4	0.00000	0.9752	0.9255	0.9983
75	5	0.00000	0.9748	0.9254	0.9981
74	6	0.00000	0.9755	0.9257	0.9989
73	32	0.00000	0.9737	0.9256	0.9981
72	34	0.00000	0.9738	0.9255	0.9986
71	35	0.00000	0.9744	0.9257	0.9981
70	36	0.00000	0.9766	0.9255	0.9988
69	38	0.00000	0.9755	0.9254	0.9986
68	56	0.00000	0.9770	0.9255	0.9945
67	57	0.00000	0.9758	0.9256	0.9984
66	58	0.00000	0.9768	0.9256	0.9987
65	59	0.00000	0.9747	0.9253	0.9985
64	60	0.00000	0.9736	0.9255	0.9983
63	61	0.00000	0.9745	0.9260	0.9987
62	62	0.00000	0.9742	0.9255	0.9983
61	63	0.00000	0.9769	0.9253	0.9985
60	64	0.00000	0.9743	0.9253	0.9984
59	65	0.00000	0.9772	0.9255	0.9984
58	66	0.00000	0.9723	0.9256	0.9984
57	69	0.00000	0.9719	0.9262	0.9989
56	70	0.00000	0.9715	0.9257	0.9983
55	72	0.00003	0.9726	0.9257	0.9990
54	73	0.00003	0.9721	0.9257	0.9984
53	33	0.00004	0.9736	0.9259	0.9986
52	50	0.00004	0.9722	0.9258	0.9985
51	20	0.00005	0.9701	0.9255	0.9990
50	46	0.00010	0.9714	0.9259	0.9988
49	51	0.00010	0.9719	0.9256	0.9982
48	15	0.00012	0.9707	0.9257	0.9985
47	26	0.00020	0.9724	0.9256	0.9987
46	8	0.00024	0.9719	0.9258	0.9986
45	17	0.00029	0.9720	0.9254	0.9988
44	16	0.00069	0.9704	0.9253	0.9983
43	27	0.00074	0.9717	0.9256	0.9986
42	37	0.00077	0.9715	0.9255	0.9653
41	25	0.00107	0.9721	0.9255	0.9991
40	22	0.00113	0.9744	0.9256	0.9654
39	74	0.00141	0.9750	0.9259	0.9659
38	71	0.00170	0.9738	0.9248	0.9991
37	30	0.00258	0.9756	0.9251	0.9992

36	10	0.00340	0.9728	0.9255	0.9994
35	7	0.00379	0.9741	0.9255	0.9992
34	9	0.00379	0.9750	0.9258	0.9973
33	54	0.00379	0.9743	0.9254	0.9993
32	52	0.00544	0.9631	0.9252	0.9948
31	67	0.00622	0.9590	0.9255	0.9973
30	76	0.00727	0.9590	0.9255	0.9950
29	29	0.00754	0.9592	0.9251	0.9990
28	31	0.00864	0.9591	0.9255	0.9995
27	45	0.00864	0.9601	0.9250	0.9996
26	68	0.01153	0.9594	0.9248	0.9957
25	48	0.01517	0.9615	0.9251	0.9995
24	28	0.01995	0.9573	0.9252	0.9996
23	44	0.02789	0.9585	0.9250	0.9997
22	47	0.02991	0.9355	0.9249	0.9996
21	49	0.03149	0.9056	0.9252	0.9999
20	1	0.03463	0.9031	0.9252	0.9988
19	2	0.04533	0.9015	0.9253	0.9987
18	21	0.04604	0.9001	0.9253	0.9989
17	78	0.09395	0.9006	0.9259	0.9986
16	19	0.09902	0.9034	0.9264	0.9986
15	24	0.09947	0.9013	0.9267	0.9987
14	75	0.09963	0.8995	0.9265	0.9987
13	77	0.10256	0.8948	0.9265	0.9990
12	18	0.11219	0.8928	0.9268	0.9989
11	43	0.11705	0.8912	0.9270	0.9992
10	23	0.13263	0.8883	0.9269	0.9991
9	12	0.13828	0.8884	0.9271	0.9990
8	41	0.18091	0.8883	0.9270	0.9992
7	53	0.18388	0.8882	0.9271	0.9992
6	39	0.19795	0.8882	0.9271	0.9989
5	55	0.21179	0.8881	0.9271	0.9989
4	13	0.21180	0.8881	0.9258	0.9988
3	40	0.21397	0.8881	0.9259	0.9985
2	42	0.23147	0.8875	0.9023	0.9994

4.2. Multiclass Classification

For multiclass classification, information packets that represent normal network traffic from both datasets are deleted and only the attack information packets are kept. with 321,283 tuples that represent packet information of nine different types of attacks in UNSW-NB15 and 557646 tuples in CICIDS2017 datasets that represent packet information of fourteen different types of attacks. GBT classifier is not used for multiclass classification because Apache Spark does not support it.

4.2.1. UNSW-NB15 Dataset

321,283 tuples using for multi-class classification that represent packet information of nine different types of attacks in UNSW-NB15. Table 4.23 showing the summarized results of two classifier DNN. using the proposed approach to feature selecting, the results show an improvement in the accuracy of the DNN classifiers. Table 4.24 shows the confusion matrix of the predictions provided by the DNN classifier used in this experiment.

Table 4.23: Result of DNN multiclass classification with UNSW-NB15 dataset

	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.01	5.15	97.04	4.71	29

In this experiment, an average of 24.85 ns is consumed by the RF classifier to predict one of the nine attack labels for each tuple and the accuracy rate is 91.76%. The subset of 29 features selected by using feature selection approach are achieved an accuracy of 91.77% and prediction time 23.63 ns. Table 4.25 shows the confusion matrix of the predictions provided by the classifier used in this experiment.

Table 4.24: Result RF of multiclass classification with UNSW-NB15 dataset

	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
RF	91.76%	24.85	91.77%	23.63	29

Table 4.25: Confusion matrix of the DNN multiclass classification with UNSW-NB15.

		Predicted								
		Analysis	Backdoors	DoS	Exploits	Fuzzers	Generic	Reconnaissance	Shellcode	Worms
Actual	Analysis	1350	0	64	938	285	16	24	0	0
	Backdoors	0	1549	124	165	348	16	127	0	0
	DoS	11	0	14776	42	649	390	428	57	0
	Exploits	93	0	631	43036	84	561	68	52	0
	Fuzzers	29	0	67	790	22656	107	565	32	0
	Generic	29	208	98	359	408	214165	183	30	1
	Reconnaissance	9	0	130	73	524	25	13226	0	0
	Shellcode	0	0	0	288	221	16	54	932	0
	Worms	2	0	0	27	16	24	0	1	104

Table 4.26: Confusion matrix of the RF multiclass classification with UNSW-NB15.

		Predicted								
		Analysis	Backdoors	DoS	Exploits	Fuzzers	Generic	Reconnaissance	Shellcode	Worms
Actual	Analysis	596	0	592	1243	223	23	0	0	0
	Backdoors	0	350	601	1141	226	7	4	0	0
	DoS	0	0	8366	7712	231	22	19	2	1
	Exploits	0	2	6148	37801	472	64	32	3	3
	Fuzzers	2	2	613	1675	21936	9	9	0	0
	Generic	10	0	631	2524	5	212304	4	3	0
	Reconnaissance	1	1	743	1425	7	5	11805	0	0
	Shellcode	0	0	1	0	1	0	0	1509	0
	Worms	0	0	1	10	0	0	0	0	163

Table 4.27: Results of feature ranking and M-Class classification on UNSW-NB15

Features	Removed Feature	F-Rank	DNN-ACC	RF-ACC
43			0.970176	0.9176
42	32	0.00000	0.969858	0.9176
41	22	0.00000	0.970432	0.9176
40	4	0.00008	0.969808	0.9176
39	8	0.00013	0.970264	0.9176
38	5	0.00024	0.970027	0.9176
37	35	0.00024	0.970458	0.9176
36	36	0.00033	0.970387	0.9176
35	9	0.00050	0.969518	0.9175
34	27	0.00054	0.970013	0.9175
33	14	0.00058	0.969926	0.9175
32	13	0.00068	0.969854	0.9176
31	26	0.00165	0.969613	0.9177
30	12	0.00180	0.970438	0.9177
29	25	0.00181	0.970464	0.9177
28	24	0.00183	0.970041	0.9175
27	21	0.00205	0.970099	0.9175
26	34	0.00208	0.970192	0.9175
25	23	0.00721	0.970076	0.9175
24	6	0.00750	0.969492	0.9175
23	28	0.00784	0.968671	0.9175
22	3	0.00797	0.969088	0.9175
21	30	0.00798	0.969155	0.9175
20	17	0.01385	0.969102	0.9175
19	18	0.03947	0.968871	0.9175
18	31	0.05884	0.968506	0.9175
17	16	0.06158	0.969479	0.9175
16	15	0.06166	0.969028	0.9174
15	7	0.07820	0.968624	0.9174
14	29	0.08158	0.968287	0.9174
13	2	0.10041	0.967739	0.9169
12	33	0.12162	0.967302	0.9152
11	20	0.12162	0.965938	0.8484
10	11	0.13263	0.965853	0.8484
9	1	0.13351	0.965885	0.8484
8	19	0.14530	0.965275	0.8455
7	40	0.14943	0.965441	0.8408
6	39	0.15862	0.964903	0.8361
5	43	0.15960	0.965087	0.8290
4	37	0.16260	0.964236	0.8149
3	41	0.16462	0.963745	0.7951
2	38	0.18168	0.962656	0.7702

4.2.2. CICIDS2017 Dataset

This experiment using 557,646 tuples represent packet information of fourteen different types of attacks in CICIDS2017 datasets. The dataset using by tow method as in the preview's experiments, which Rep-CICIDS2017 and Rem-CICIDS2017.

According to the results shown in Table 4.28, the performance of the DNN classifier achieved a high accuracy rate of 99.55% on Rep-CICIDS2017. While the accuracy of the DNN classifier improved slightly when implemented on a subset of 75 features. Table 4.30 illustrates the confusion matrix.

Table 4.28: Result of DNN multiclass classification with Rep-CICIDS2017 dataset

Replace					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99.55	0.64	99.57	0.70	75

As in Table 4.29, the average prediction time of the DNN classifier in this experiment is 0.63 ns per each tuple of Rep-CICIDS2017, and the accuracy is 99.56%, while the average of 0.73 ns is consumed by the classifier on the subset of 72 features to predict one of the nine attack labels for each tuple. Table 4.31 shows the confusion matrix of DNN.

Table 4.29: Result of DNN multiclass classification with Rem-CICIDS2017 dataset

Remove					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99.56%	0.63	99.56%	0.73	67

Table 4.30: Confusion matrix of the DNN multi-class classification with Rep-CICIDS2017

	Predicted														
Actual	**Replace**	**Bot**	**DDoS**	**DoS GoldenEye**	**DoS Hulk**	**DoS Slowhttptest**	**DoS slowloris**	**FTP-Patator**	**Heartbleed**	**Infiltration**	**PortScan**	**SSH-Patator**	**Web Attack Brute Force**	**Web Attack Sql Injection**	**Web Attack XSS**
	Bot	1869	0	0	25	1	1	0	0	0	6	64	0	0	0
	DDoS	3	127974	13	20	0	0	0	0	0	17	0	0	0	0
	DoS GoldenEye	0	142	9992	72	4	7	0	0	0	0	76	0	0	0
	DoS Hulk	121	13	5	230887	6	0	0	0	0	11	27	3	0	0
	DoS Slowhttptest	1	1	3	2	5272	159	1	0	0	27	22	11	0	0
	DoS slowloris	0	0	38	4	376	4732	522	0	0	9	82	33	0	0
	FTP-Patator	0	0	0	11	0	2	7918	0	0	0	6	1	0	0
	Heartbleed	0	0	1	10	0	0	0	0	0	0	0	0	0	0
	Infiltration	0	1	2	1	13	11	0	0	6	1	1	0	0	0
	PortScan	0	3	12	13	3	44	1	0	0	158801	44	9	0	0
	SSH-Patator	1	0	0	9	0	2	27	0	0	19	5839	0	0	0
	Web Attack Brute Force	0	0	75	1	0	0	2	0	0	3	16	1410	0	0
	Web Attack Sql Injection	0	0	5	0	0	0	1	0	0	0	11	0	4	0
	Web Attack XSS	0	0	0	17	0	0	0	0	0	1	9	101	0	524

Table 4.31: Confusion matrix of the DNN multi-class classification with Rem-CICIDS2017

	Predicted														
	Remove	**Bot**	**DDoS**	**DoS GoldenEye**	**DoS Hulk**	**DoS Slowhttptes t**	**DoS slowloris**	**FTP-Patator**	**Heartbleed**	**Infiltration**	**PortScan**	**SSH-Patator**	**Web Attack Brute Force**	**Web Attack Sql Injection**	**Web Attack XSS**
Actual	**Bot**	1868	0	0	16	1	1	0	0	0	6	64	0	0	0
	DDoS	3	127973	13	19	0	0	0	0	0	17	0	0	0	0
	DoS GoldenEye	0	142	9992	72	4	7	0	0	0	0	76	0	0	0
	DoS Hulk	138	13	5	229921	6	0	0	0	0	11	27	3	0	0
	DoS Slowhttptest	1	1	3	2	5272	159	1	0	0	27	22	11	0	0
	DoS slowloris	0	0	38	4	376	4732	522	0	0	9	82	33	0	0
	FTP-Patator	0	0	0	11	0	2	7915	0	0	0	6	1	0	0
	Heartbleed	0	0	1	10	0	0	0	0	0	0	0	0	0	0
	Infiltration	0	1	2	1	13	11	0	0	6	1	1	0	0	0
	PortScan	0	3	12	13	3	44	1	0	0	158675	44	9	0	0
	SSH-Patator	1	0	0	9	0	2	27	0	0	19	5839	0	0	0
	Web Attack Brute Force	0	0	75	1	0	0	2	0	0	3	16	1410	0	0
	Web Attack Sql Injection	0	0	5	0	0	0	1	0	0	0	11	0	4	0
	Web Attack XSS	0	0	0	17	0	0	0	0	0	1	9	101	0	524

In this experiment, the RF classifier applied on the Rep-CICIDS to classify tuples into one of the fourteen different attacks. Table 4.32 shows the accuracy rate and prediction time, where the classifier achieved an accuracy of 92.57% and prediction time is 7.40 ns. Eight features selected that achieved the best accuracy of 92.72% with slight improvement. The predictions provided by the classifier are compared to the actual labels of these tuples in the confusion matrix shown in Table 4.34.

Table 4.32: Result of RF multiclass classification with Rep-CICIDS2017 dataset

Replace					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
RF	92.57%	7.40	92.72%	6.22	8

The Rep-CICIDS are used to evaluate the RF classifier with the feature selection technique, the Table 4.33 shows the accuracy rate is 92.54% when implemented the classifier on full dataset, and there is marginal improvement by using the subset of 6 features that obtain the high homogeneity rank, where the classifier achieved an accuracy rate is 92.71%. Table 4.35 shows the confusion matrix.

Table 4.33: Result of RF multiclass classification with Rem-CICIDS2017 dataset

Remove					
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
RF	92.54%	6.98	92.71%	6.04	6

The results of features ranking and accuracy rate of the DNN and RF classifiers those implemented on Rep-CICIDS2017 and Rem-CICIDS2017 datasets are showing in Table 4.36 and Table 4.37, respectively.

Table 4.34: Confusion matrix of the RF multi-class classification with Rep-CICIDS2017

	Predicted													
Actual	Replace	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	Heartbleed	Infiltration	PortScan	SSH-Patator	Web Attack Brute Force	Web Attack Sql Injection	Web Attack XSS
	Bot	0	736	738	2	0	0	0	0	0	490	0	0	0	0
	DDoS	69	127913	0	29	0	0	0	0	0	0	16	0	0	0
	DoS GoldenEye	0	0	0	5371	998	2280	75	0	350	108	0	88	489	534
	DoS Hulk	0	15	121	230583	206	2	145	0	0	1	0	0	0	0
	DoS Slowhttptest	0	224	2110	356	0	0	0	1	0	2808	0	0	0	0
	DoS slowloris	0	1768	2309	80	0	0	0	0	0	1639	0	0	0	0
	FTP-Patator	0	3953	0	17	0	0	0	0	0	3968	0	0	0	0
	Heartbleed	0	8	0	0	1	0	1	0	0	1	0	0	0	0
	Infiltration	0	2	33	0	0	0	0	0	0	1	0	0	0	0
	PortScan	29	0	40	60	255	1	0	0	0	158542	3	0	0	0
	SSH-Patator	0	17	2939	8	0	0	0	0	0	2933	0	0	0	0
	Web Attack Brute Force	0	0	216	74	0	0	0	0	0	1217	0	0	0	0
	Web Attack Sql Injection	0	0	14	2	0	0	0	0	0	5	0	0	0	0
	Web Attack XSS	0	0	43	2	0	0	0	0	0	607	0	0	0	0

Table 4.35: Confusion matrix of the RF multi-class classification with Rem-CICIDS2017

		Predicted													
	Remove	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	Heartbleed	Infiltration	PortScan	SSH-Patator	Web Attack Brute Force	Web Attack Sql Injection	Web Attack XSS
Actual	Bot	0	726	738	2	0	0	0	0	0	490	0	0	0	0
	DDoS	68	127900	0	29	0	12	0	0	0	0	16	0	0	0
	DoS GoldenEye	0	0	0	5372	1083	2283	75	0	350	43	0	91	481	515
	DoS Hulk	0	17	115	229638	205	2	145	0	0	1	0	1	0	0
	DoS Slowhttptest	0	224	2110	356	0	0	0	1	0	2808	0	0	0	0
	DoS slowloris	0	1768	2310	80	0	0	0	0	0	1638	0	0	0	0
	FTP-Patator	0	7905	0	17	0	0	0	0	0	13	0	0	0	0
	Heartbleed	0	9	0	0	1	0	0	0	0	1	0	0	0	0
	Infiltration	0	2	33	0	0	0	0	0	0	1	0	0	0	0
	PortScan	29	0	40	60	243	1	0	0	0	158428	3	0	0	0
	SSH-Patator	0	19	2950	8	0	0	0	0	0	2920	0	0	0	0
	Web Attack Brute Force	0	0	218	73	0	0	0	0	0	1216	0	0	0	0
	Web Attack Sql Injection	0	0	14	2	0	0	0	0	0	5	0	0	0	0
	Web Attack XSS	0	0	44	2	0	0	0	0	0	606	0	0	0	0

Table 4.36: The results of feature ranking and multi-class classification with Rep-CICIDS2017

Features	Removed Feature	F-Rank	DNN-ACC	RF-ACC
78			0.9955	0.9257
77	32	0	0.9955	0.9259
76	33	0	0.9954	0.9255
75	34	0	0.9957	0.9254
74	46	0	0.9946	0.9254
73	50	0	0.9952	0.9256
72	51	0	0.9956	0.9255
71	57	0	0.9955	0.9253
70	58	0	0.9950	0.9258
69	59	0	0.9949	0.9254
68	60	0	0.9951	0.9256
67	61	0	0.9956	0.9255
66	62	0	0.9956	0.9254
65	72	0.01188	0.9955	0.9251
64	15	0.0163	0.9954	0.9254
63	30	0.02156	0.9953	0.9255
62	49	0.02239	0.9946	0.9256
61	31	0.03202	0.9942	0.9254
60	45	0.03202	0.9936	0.9258
59	20	0.0382	0.9921	0.9254
58	74	0.03979	0.9933	0.9258
57	25	0.04715	0.9913	0.9259
56	71	0.04718	0.9922	0.9257
55	73	0.0495	0.9927	0.9258
54	76	0.05322	0.9930	0.9256
53	44	0.07368	0.9912	0.9258
52	27	0.11298	0.9911	0.9254
51	48	0.16125	0.9920	0.9255
50	28	0.17754	0.9906	0.9254
49	52	0.20243	0.9920	0.9256
48	26	0.2039	0.9908	0.9257
47	29	0.20659	0.9911	0.9256
46	14	0.21546	0.9896	0.9257
45	6	0.21685	0.9903	0.9256
44	66	0.21685	0.9904	0.9259
43	8	0.23151	0.9910	0.9255
42	43	0.23777	0.9901	0.9258
41	39	0.24161	0.9899	0.9255
40	5	0.25238	0.9905	0.9259

39	64	0.25243	0.9908	0.9256
38	40	0.25285	0.9898	0.9253
37	4	0.25811	0.9899	0.9254
36	65	0.25811	0.9892	0.9255
35	22	0.26455	0.9897	0.9251
34	42	0.26923	0.9896	0.9256
33	47	0.28221	0.9863	0.9257
32	11	0.28657	0.9862	0.9254
31	17	0.28842	0.9858	0.9252
30	78	0.29142	0.9859	0.9256
29	41	0.30188	0.9859	0.9251
28	53	0.30296	0.9862	0.9254
27	77	0.30385	0.9856	0.9252
26	18	0.30606	0.9848	0.9252
25	75	0.30758	0.9853	0.9253
24	23	0.31161	0.9838	0.9252
23	1	0.33756	0.9816	0.9252
22	21	0.34186	0.9827	0.9254
21	69	0.35098	0.9808	0.9252
20	24	0.35774	0.9823	0.9251
19	3	0.36944	0.9806	0.9254
18	63	0.36944	0.9810	0.9259
17	2	0.37961	0.9808	0.9260
16	19	0.38341	0.9756	0.9264
15	36	0.38887	0.9777	0.9265
14	10	0.40721	0.9681	0.9264
13	37	0.41205	0.9588	0.9268
12	38	0.41221	0.9624	0.9267
11	16	0.41843	0.9597	0.9270
10	12	0.42154	0.9567	0.9269
9	7	0.44074	0.9565	0.9269
8	68	0.451	0.9538	0.9272
7	35	0.49988	0.9514	0.9272
6	56	0.50071	0.9544	0.9272
5	13	0.51447	0.9574	0.9271
4	55	0.51447	0.9527	0.9258
3	9	0.5455	0.9481	0.9259
2	54	0.54697	0.9511	0.9024
			0.9957	0.9272

Table 4.37: Results of feature ranking and multi-class classification with Rem-CICIDS2017

Features	Removed Feature	Rank	DNN-ACC	RF-ACC
78			0.9956	0.9254
76	33	0	0.9954	0.9254
75	34	0	0.995642	0.9255
74	46	0	0.9945	0.9252
73	50	0	0.9951	0.9254
72	51	0	0.9956	0.9253
71	57	0	0.9953	0.9250
70	58	0	0.9947	0.9251
69	59	0	0.9949	0.9251
68	60	0	0.9953	0.9253
67	61	0	0.9956	0.9253
66	62	0	0.9955	0.9256
65	72	0	0.9955	0.9251
64	15	0.01189	0.9954	0.9253
63	30	0.0209	0.9954	0.9254
62	49	0.02159	0.9945	0.9252
61	31	0.02239	0.9941	0.9253
60	45	0.03203	0.9938	0.9255
59	20	0.03203	0.9921	0.9254
58	74	0.03823	0.9935	0.9254
57	25	0.03982	0.9912	0.9254
56	71	0.04717	0.9922	0.9255
55	73	0.04721	0.9927	0.9258
54	76	0.04956	0.9933	0.9253
53	44	0.05325	0.9911	0.9256
52	27	0.07399	0.9909	0.9257
51	48	0.11325	0.9921	0.9253
50	28	0.16082	0.9906	0.9253
49	52	0.17624	0.9920	0.9256
48	26	0.20212	0.9908	0.9257
47	29	0.2065	0.9912	0.9252
46	6	0.20974	0.9893	0.9252
45	66	0.2177	0.9903	0.9255
44	14	0.2177	0.9909	0.9254
43	8	0.21802	0.9911	0.9256
42	43	0.23145	0.9903	0.9255
41	39	0.23907	0.9899	0.9257
40	64	0.24155	0.9905	0.9255
39	5	0.25364	0.9909	0.9256
38	40	0.25365	0.9898	0.9255
37	4	0.25373	0.9899	0.9252
36	65	0.25899	0.9892	0.9256
35	42	0.25899	0.9899	0.9253

34	22	0.26785	0.9899	0.9252
33	47	0.27328	0.9862	0.9254
32	17	0.28179	0.9862	0.9253
31	11	0.28275	0.9859	0.9254
30	78	0.28751	0.9856	0.9253
29	75	0.29226	0.9858	0.9249
28	41	0.30213	0.9862	0.9250
27	53	0.30281	0.9856	0.9251
26	77	0.30391	0.9847	0.9248
25	18	0.30458	0.9853	0.9252
24	23	0.30698	0.9838	0.9249
23	21	0.31105	0.9814	0.9254
22	1	0.3361	0.9830	0.9250
21	69	0.33748	0.9808	0.9250
20	24	0.34746	0.9823	0.9250
19	3	0.35563	0.9806	0.9254
18	63	0.36749	0.9812	0.9251
17	2	0.36749	0.9807	0.9258
16	19	0.38546	0.9757	0.9263
15	36	0.38707	0.9775	0.9262
14	10	0.38997	0.9680	0.9262
13	38	0.40827	0.9585	0.9262
12	37	0.41361	0.9624	0.9266
11	16	0.41377	0.9597	0.9269
10	12	0.41827	0.9568	0.9269
9	7	0.4215	0.9565	0.9268
8	68	0.44367	0.9538	0.9270
7	35	0.45219	0.9515	0.9270
6	56	0.49833	0.9543	0.9271
5	13	0.49862	0.9573	0.9270
4	55	0.51625	0.9527	0.9257
3	54	0.51625	0.9481	0.9258
2	9	0.54662	0.9512	0.9023
		0.54669	0.9956	0.9271

CHAPTER 5

DISCUSSION

In binary classification with UNSW-NB15 dataset, although the differences are marginal. As in Table 5.1 the results show the accuracies are provided by the proposed method, where the three classifiers achieved high accuracy rate. The DNN with three hidden layers is the best classifier where achieved the best accuracy rate with 41 features compared to the rest of the classifiers, is 99.19%. The GBT provided an extremely lower average prediction time per each packet, which is 0.35 ns, compared to the DNN and RF classifiers, which consumed 1.35 ns and 11.16 ns respectively. Moreover, by comparing the results of the proposed method with the results of classifiers on full dataset, there was a slight improvement in the accuracy and time prediction.

Table 5.1: Summary of binary classification with UNSW-NB15 dataset

Classifier	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99.16%	1.43	99.19%	1.35	41
RF	98.85%	12.13	98.86%	11.36	32
GBT	97.83%	0.70	97.92%	0.35	26

Using a CICIDS2017 data set with two methods that replacing and removing the massing and infinity values for binary classification. The best accuracy rate provided by the GBT classifier on the 23 features selected from the Rep-CICIDS2017, which is 99.81%. The DNN classifier achieved the lower time prediction, which consumed 0.05 ns and a lower accuracy rate than GBT, which is 92.72%. Table 5.2 shows the differences are marginal, here there is an improvement in the accuracy of the works with the suggested approach, although small compared with the results of the full dataset.

Table 5.2: Summary of binary classification with Rep-CICIDS2017 dataset

Classifier	Replace				
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.72%	0.05	97.73%	0.05	59
RF	92.54%	7.71	92.72%	0.18	6
GBT	99.81%	1.23	99.97%	0.01	23

According to the Table 5.3 shows the results of the implementation of the proposed method on the Rem-CICIDS2017 data set, there is a very small difference in the results. The RF achieved the lowest rate of performance and the highest consumer time to prediction when compared to other classifiers.

Table 5.3: Summary of binary classification with Rem-CICIDS2017 dataset

Classifier	Remove				
	Accuracy – full dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.71%	0.05	97.72%	0.05	59
RF	92.54%	7.68	92.71%	0.13	8
GBT	99.81%	1.24	99.99%	0.03	21

For the multi-class attacks prediction, the results show the superiority of the DNN using the UNSW-NB15 attacks packets, which consumed 5.15 ns and 4.71 ns, while provided accuracy 97.01%, and 97.04 by using the homogeneity metric for feature selection. While the RF achieved an accuracy of 91.76% with UNSW-NB15 attacks packets and 91.77% with the subset of 31 features selection. In contrast, the results of the proposed feature selection method indicate an incommodious improvement in the accuracy rate and prediction time.

Table 5.4: Summary of multiclass classification with UNSW-NB15 dataset

Classifier	Accuracy – attack dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	97.01%	5.15	97.04%	4.71	29
RF	91.76%	24.85	91.77%	23.48	31

By using attack packets of Rep-CICIDS2017, DNN achieved a high accuracy rate in the classification of attack packets, which is 99.55% and prediction time is 0.64 ns. The DNN classifier obtained the same accuracy on subset of 75 features with very little improvement. The RF classifier achieved a lower accuracy rate with attack packets compared to DNN, also with a subset of features selected from the entire attack packets.

Table 5.5: Summary of multiclass classification with Rep-CICIDS2017 dataset

Classifiers	Rep-CICIDS2017				
	Accuracy – attack dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99.55%	0.64	99.57%	0.70	75
RF	92.57%	7.40	92.72%	6.22	8

As in Table 5.6, there is no significant difference in the applied of classifiers to the attack data packets of Rem-CICIDS2017 data. This also applies to the time of expectation, which indicates that there is little effect either by deleting the lost and infinite values or replacing them with the mean values.

Table 5.6: Summary of multiclass classification with Rem-CICIDS2017 dataset

Classifier	Rem-CICIDS2017				
	Accuracy – attack dataset	P. Time (ns)	Best accuracy with FS	P. Time (ns)	Number of Features
DNN	99.56%	0.63	99.56%	0.85	67
RF	92.54%	6.98	92.71%	6.04	6

In general, the homogeneity metric is a simple and uncomplicated method. This proposed technique has achieved a high accuracy rate with fewer features when compared with the applied of classifiers on full two datasets. as in Figure 5.1 and Figure 5.2, Note that the curves maintain the same level almost with features that have a lower rank and then starts to decline with the remove features that have a higher rank. The slight improvement in classifier performance using the proposed feature selection approach suggests that this approach can be developed to deal more efficiently with heterogeneous data, which is one of the most significant challenges of intrusion detection systems.

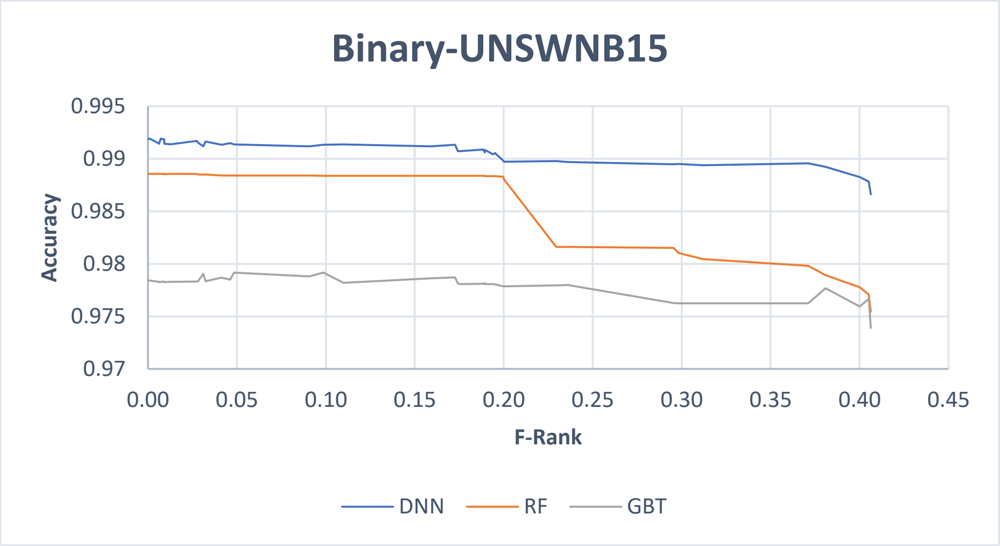

Figure 5.1: Correlation of feature rank and accuracy

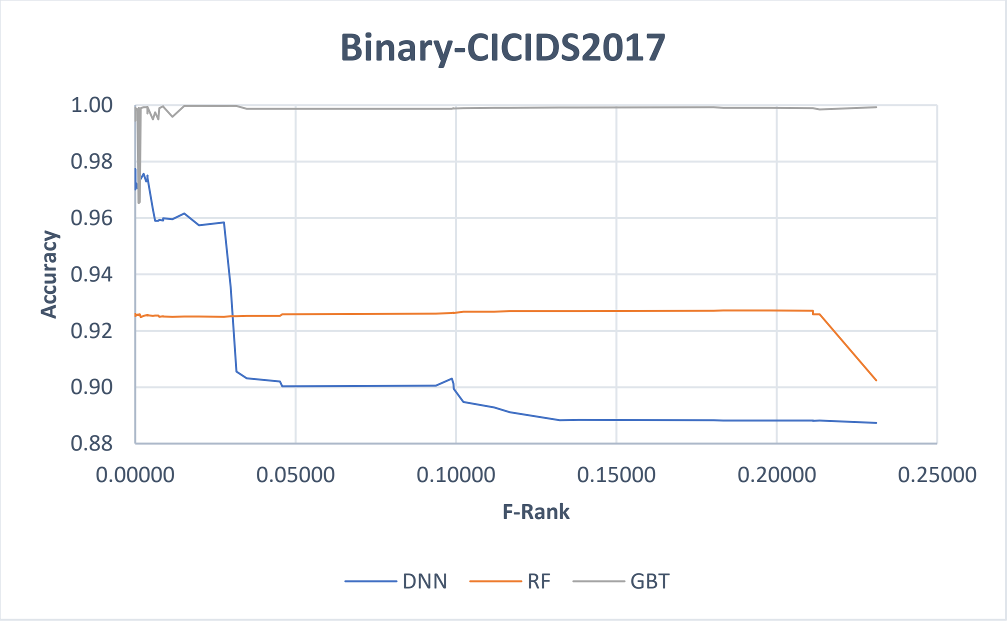

Figure 5.2: Correlation of feature rank and accuracy

Figure 5.3 Illustration that DNN remains at the same accuracy with marginal improvement and a slight slope whenever high homogeneity features are removed. The

RF is more steeply slanted in accuracy whenever the range of features that achieve high homogeneity. As in Figure 5.4 shows the DNN obtained the high accuracy rate on full features of CICIDS2017 dataset. The curve maintains the same accuracy rate with features that have the least homogeneity rank and then starts gradual regression whenever the higher features are removed. The RF maintains a constant accuracy rate until the last feature is reached.

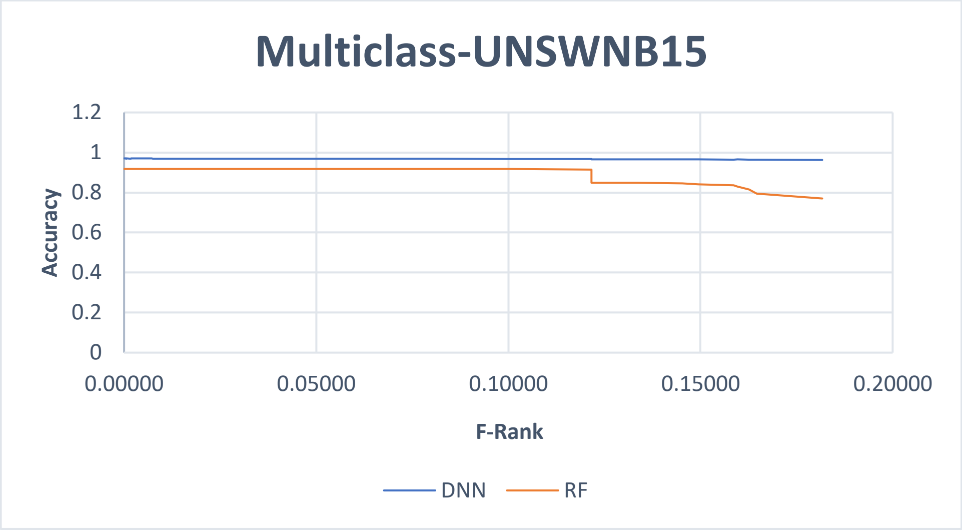

Figure 5.3: Correlation of feature rank and accuracy

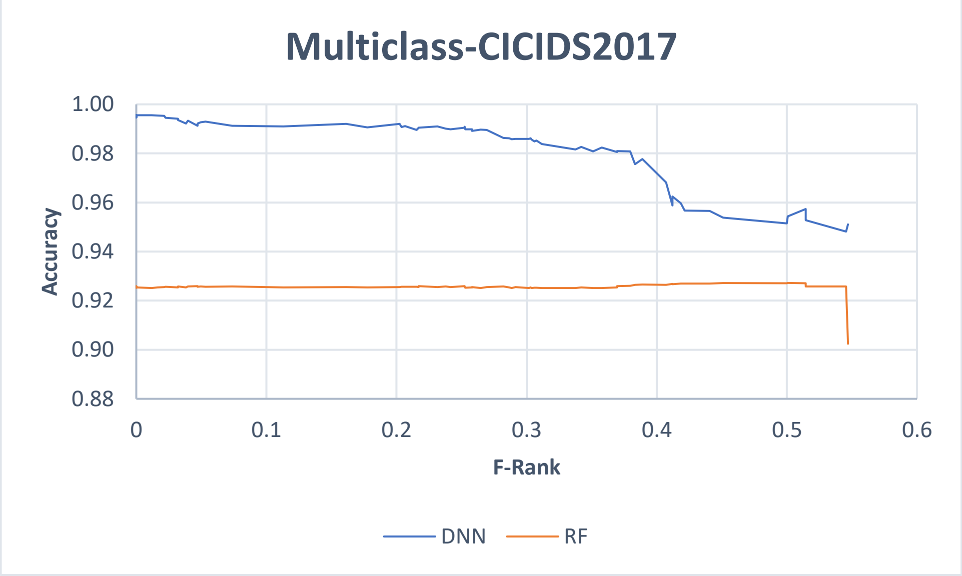

Figure 5.4: Correlation of feature rank and accuracy

In binary classification scenario, The DDN and GBT classifiers achieved best accuracy and time prediction. While the RF classifier came at a lower rate compared to other classifiers. Given the overall performance of deep learning and ensemble techniques, they have the ability to handle large data and improve the performance of intrusion detection systems in detecting abnormal patterns in data with less false alarm rate. For multiclass classification, The DNN classifier achieved a high accuracy rate, which reduced the false rate alarms while the random forest classifier gave a lower accuracy compared to the two.

Two recent sets of data were used, including a set of common attacks to evaluate the proposed method, A UNSW-NB15 dataset has been used in many previous studies and has proved, as in this study, that it has the efficiency to be used to evaluate modern approaches aimed at detecting modern and sophisticated patterns of attacks. A CICIDS2017 dataset contains a wider range of common and modern attacks than large-scale datasets, in this study, the results showed that there is no significant effect on the performance of the classifiers because the size of the data that have missing or infinite values is small compared to the total size of the CICIDS2017 dataset.

The use of apache spark technique has greatly improved the prediction time of the three classifiers when compared to traditional techniques, this improvement gives intrusion detection systems the ability to make decisions more efficiently in terms of blocking or allowing data to pass through the network, in addition, the integration between Apache Spark and the Keras Deep Learning Library has increased the capabilities of deep learning algorithms to work more efficiently and quickly.

As shown in Table 1.7, the accuracy of classifiers used in this study is compared with previous studies in binary classification, while table 1.8 shown the comparison in multiclass classification.

Table 5.7: Comparison of the accuracy of the binary classification prediction with earlier studies.

Study	Classifier	Accuracy (%)
Primartha and Tama [62]	Random Forest	95.5
	Multilayer Perceptron	83.50
Nour Mustafa and Jill Slay [35]	Naïve Bayes	79.50
	Expectation-Maximization	77.20
	Linear Regression	83.00
Mustapha Belouch, et al. [79]	RepTree	87.80
	Naïve Bayes	80.04
	Random Tree	86.59
	Decision Tree	86.13
	Artificial Neural Network	86.31
Malek Al-Zewairi, et al. [61]	Deep Learning	98.99
This Study	Deep neural network	**99.16**
	Random Forest	**98.85**
	Gradient Boosted Tree	**97.83**

Table 5.8: Comparison of the prediction's accuracy attack of multiclass classification with earlier studies.

Study	Classifier	Accuracy (%)
Mustapha Belouch, et al. [79]	RepTree	79.20
	Random Tree	76.21
	Naïve Bayes	73.86
	Artificial Neural Network	78.14
Hossein Gharaee, Hamid Hosseinvand [80]	Genetic + SVM	93.25
This Study	Deep neural network	**97.01**
	Random forest	**91.76**

Table 5.9 shows the comparison of classifiers accuracies rate were used in proposed method with the earlier works that used the CICIDS2017 dataset, the results show that the results are close to the simple superiority of the proposed method in this work.

Table 5.9: Comparison of the accuracy of the binary classification prediction with earlier studies used CICIDS2017 dataset.

Study	**Classifier**	**Accuracy (%)**
Iman Sharafaldin, et al. [56]	K-Nearest Neighbors	96.00
	Random Forest	98.00
	ID	98.00
	Adaboost	77.00
	Multilayer Perceptron	77.00
	Naïve Bayes	88.00
	Quadratic Discriminant Analysis	97.00
R. Vijayanand, et al. [81]	SVM+ Genetic	99.85
Alves and Drummond. [82]	Genetic + Profiling	92.85
This study	Deep neural network	97.73
	Random forest	92.72
	Gradient Boosted Tree	**99.97**

CHAPTER 6

CONCLUSION

Intrusion detection systems have been developed to protect network resources and data from any attacks that threaten their integrity and confidentiality. The emergence of the term big data over the past years, which refers to the huge amount of data that are generated daily from several sources, such as social networking sites Internet of things, factories, which constitute a real challenge to the characteristics of intrusion detection systems in monitoring network traffic, prevention and detection of attacks. The intrusion detection systems are based on two main methods of detection: misuse-based detection and anomaly-based detection. The first method is to analyze the data according to the attack knowledge databases. The second method relies on extracting abnormal patterns of data. Anomaly detection methods use machine learning techniques to mine data whose behavior differs from the behavior of normal data, often representing potential attacks.

Heterogeneity of huge amount of data is one of the barriers to intrusion detection systems for analyzing data and extracting the relevant features that represent anomaly data. Many techniques use the selection of relevant features, that help to improve the performance of classification techniques and reduce the volume of resources consumed. Several methods have been proposed to improve the performance of intrusion detection systems that have used different types of feature selection and intrusion detection techniques. Most studies rely on traditional data processing techniques to improve the performance of data intrusion detection systems which have become useless in dealing with the increasing volume of data. Studies that relied on big data techniques did not give much importance to how intrusion detection systems deal with large and heterogeneous data. Our proposed method is based on the integration of deep learning and big data techniques with a feature selection technique based on a homogeneity metric.

Our proposed method relies on the use of the homogeneity metric of the k-means clustering algorithm to select the relevant features on two recent data sets UNSW-NB15 and CICIDS2017, that contain a set of common attacks. Homogeneity is a clustering metric, where clustering must assign only those data points that are members of a single class to a single cluster where unique features can be identified for each class. The homogeneity rank of each feature is computed separately, the features are arranged in descending order from the lowest rank to the highest rank. Deep Neural Network, Random Forest and Gradient Boosting Tree classifiers are applied on the datasets with two scenarios, which are binary and multiclass classification. A feature is removed each time with the application of the classifiers on the remaining feature set until accessing to the latest feature. 5-fold cross validation method is used in all experiments to evaluate the machine learning models. The performance of the proposed method is evaluated in terms of accuracy and prediction time.

The proposed approach achieved a high-performance accuracy, as the use of the homogeneity metric of the feature rank proved that a unique feature could be identified for each class, thus increasing the performance of the classifiers in extracting anomalies in data. In binary classification, the DNN classifier achieved a high accuracy is 99.19% with a subset of 41 features when implemented on UNSW-NB15 dataset while the GBT classifier achieved the high accuracy of 99.99% with a subset of 21 features of the subset selected from CICIDS2017 dataset. For multiclass classification, the highest accuracies achieved by DNN classifier are 97.04% with the subset of 29 features and 99.56% with the subset of 67 features, that obtained the highest rank of homogeneity and selected from UNSW-NB15 and CICIDS2017 datasets respectively.

The proposed approach was compared with a set of related work that relied on the same datasets to evaluate their proposals, and the differences were marginal. Given the comparison, the proposed approach obtained a higher accuracy rate than previous work, machine learning techniques used in this proposal have the ability to deal with big data, often involving heterogeneous data, using the homogeneity metric to select a unique feature that enhances the ability of classifiers to detect anomalies. The high efficiency and speed of Apache Spark in the analysis and processing of large data and its ability to integrate with deep learning libraries improve the speed required to build models and detect anomalies, which helps to enhance the speed of decision-making and reduce the false rate alarm.

In future work, the development of homogeneity can be studied to obtain a higher homogeneity rank, and experiment with other deep learning techniques that have the ability to handle large data. Another future work could be the use of Graphics Processing Unit (GPU) for its high speed, which will reduce the training time spent on machine learning techniques.

REFERENCES

[1] Bace, R., & Mell, p. (2001). NIST special publication on intrusion detection systems. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

[2] Lazarevic, A., Kumar, V., & Srivastava, J. (2005). Intrusion detection: A survey. In Managing Cyber Threats (pp. 19-78). Springer, Boston, MA.

[3] Prasad, S. S. E., Srinath, M. V., & Basha, M. S. (2015). Intrusion detection systems, tools and techniques–an overview. Indian Journal of Science and Technology, 8(35).

[4] Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., & Atkinson, R. (2017). Shallow and deep networks intrusion detection system: A taxonomy and survey. arXiv preprint arXiv:1701.02145.

[5] Laney, D., 2001. 3D data management: Controlling data volume, velocity and variety. META group research note, 6(70), p. 1.

[6] Suthaharan, S., 2014. Big data classification: Problems and challenges in network intrusion prediction with machine learning. ACM SIGMETRICS Performance Evaluation Review, 41(4), pp. 70-73.

[7] Zuech, R., Khoshgoftaar, T.M. and Wald, R., 2015. Intrusion detection and big heterogeneous data: a survey. Journal of Big Data, 2(1), p. 3.

[8] https://www.ibmbigdatahub.com/infographic/four-vs-big-data.

[9] Demchenko, Y., De Laat, C., & Membrey, p. (2014, May). Defining architecture components of the Big Data Ecosystem. In Collaboration Technologies and Systems (CTS), 2014 International Conference on (pp. 104-112). IEEE.

[10] https://www.microsoft.com/en-us/sql-server/big-data

[11] Shvachko, Konstantin, et al. "The hadoop distributed file system." Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on. Ieee, 2010.

[12] Spark, Apache. "Apache Spark: Lightning-fast cluster computing." URL http://spark. apache. org (2016).

[13] Thusoo, A., Sarma, J. S., Jain, N., Shao, Z., Chakka, P., Anthony, S., ... & Murthy, R. (2009). Hive: a warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment, 2(2), 1626-1629.

[14] Han, J., Haihong, E., Le, G., & Du, J. (2011, October). Survey on NoSQL database. In Pervasive computing and applications (ICPCA), 2011 6th international conference on (pp. 363-366). IEEE.

[15] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., ... & Ghodsi, A. (2016). Apache spark: a unified engine for big data processing. Communications of the ACM, 59(11), 56-65.

[16] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., ... & Xin, D. (2016). Mllib: Machine learning in apache spark. The Journal of Machine Learning Research, 17(1), 1235-1241.

[17] Chollet, F. (2015). Keras: Deep learning library for theano and tensorflow. URL: https://keras. io/k, 7(8).

[18] Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.). (2013). Machine learning: An artificial intelligence approach. Springer Science & Business Media.

[19] Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. Neurocomputing, 237, 350-361.

[20] Denning, D. E. (1987). An intrusion-detection model. IEEE Transactions on software engineering, (2), 222-232.

[21] Sabahi, F., & Movaghar, A. (2008, October). Intrusion detection: A survey. In Systems and Networks Communications, 2008. ICSNC'08. 3rd International Conference on (pp. 23-26). IEEE.

[22] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 15.

[23] Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. Ieee communications surveys & tutorials, 16(1), 303-336.

[24] Agrawal, S., & Agrawal, J. (2015). Survey on anomaly detection using data mining techniques. Procedia Computer Science, 60, 708-713.

[25] Jyothsna, V. V. R. p. V., Prasad, V. R., & Prasad, K. M. (2011). A review of anomaly based intrusion detection systems. International Journal of Computer Applications, 28(7), 26-35.

[26] Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. Journal of Network and Computer Applications, 60, 19-31.

[27] Tan, M., Tsang, I. W., & Wang, L. (2014). Towards ultrahigh dimensional feature selection for big data. The Journal of Machine Learning Research, 15(1), 1371-1429.

[28] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.

[29] Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. Computers & Electrical Engineering, 40(1), 16-28.

[30] Hindy, H., Brosset, D., Bayne, E., Seeam, A., Tachtatzis, C., Atkinson, R., & Bellekens, X. (2018). A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets. arXiv preprint arXiv:1806.03517.

[31] Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), 100-108.

[32] Rosenberg, A., & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL).

[33] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_score.html

[34] Nisioti, A., Mylonas, A., Yoo, p. D., & Katos, V. (2018). From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods. IEEE Communications Surveys & Tutorials.

[35] Moustafa, N., & Slay, J. (2017). A hybrid feature selection for network intrusion detection systems: Central points. arXiv preprint arXiv:1707.05505.

[36] Hajisalem, V., & Babaie, S. (2018). A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. Computer Networks, 136, 37-50.

[37] Idhammad, M., Afdel, K., & Belouch, M. (2017). Dos detection method based on artificial neural networks. Int J Adv Comput Sci Appl (ijacsa), 8(4), 465-471.

[38] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural networks, 61, 85-117.

[39] K. Gurney, (2014) An introduction to neural networks: CRC press.

[40] https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[41] B. Xu, N. Wang, T. Chen, and M. Li,Â (2015 of Conference) "Empirical evaluation of rectified activations in convolutional network," arXiv preprint arXiv:1505.00853.

[42] Buscema, M. (1998). Back propagation neural networks. Substance use & misuse, 33(2), 233-270.

[43] Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. Neurocomputing, 237, 350-361.

[44] Benmessahel, I., Xie, K., & Chellal, M. (2017). A new evolutionary neural networks based on intrusion detection systems using multiverse optimization. Applied Intelligence, 1-13.

[45] Guha, S., Yau, S. S., & Buduru, A. B. (2016, August). Attack detection in cloud infrastructures using artificial neural network with genetic feature selection. In Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2016 IEEE 14th Intl C (pp. 414-419). IEEE.

[46] Nguyen, K. K., Hoang, D. T., Niyato, D., Wang, P., Nguyen, D., & Dutkiewicz, E. (2018, April). Cyberattack detection in mobile cloud computing: A deep learning approach. In Wireless Communications and Networking Conference (WCNC), 2018 IEEE (pp. 1-6). IEEE.

[47] Muna, A. H., Moustafa, N., & Sitnikova, E. (2018). Identification of malicious activities in industrial internet of things based on deep learning models. Journal of Information Security and Applications, 41, 1-11.

[48] Zhou, Z. H. (2012). Ensemble methods: foundations and algorithms. Chapman and Hall/CRC.

[49] https://www.linkedin.com/pulse/random-forest-algorithm-interactive-discussion-niraj-kumar/

[50] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[51] Liu, Y. (2014). Random forest algorithm in big data environment. Computer Modelling & New Technologies, 18(12A), 147-151.

[52] https://spark.apache.org/docs/latest/mllib-ensembles.html#classification

[53] Malik, A. J., Shahzad, W., & Khan, F. A. (2015). Network intrusion detection using hybrid binary PSO and random forests algorithm. Security and Communication Networks, 8(16), 2646-2660.

[54] Farnaaz, N., & Jabbar, M. A. (2016). Random forest modeling for network intrusion detection system. Procedia Computer Science, 89, 213-217.

[55] Shi, N., Yuan, X., & Nick, W. (2017). Semi-supervised Random Forest for Intrusion Detection Network.

[56] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018, January). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In ICISSP (pp. 108-116).

[57] Friedman, J. H. (2002). Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367-378.

[58] Tama, Bayu Adhi, and Kyung-Hyune Rhee (2017). "An in-depth experimental study of anomaly detection using gradient boosted machine." Neural Computing and Applications: 1-11.

[59] Gupta, G. P., & Kulariya, M. (2016). A framework for fast and efficient cyber security network intrusion detection using apache spark. Procedia Computer Science, 93, 824-831.

[60] Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (2012). The'K'in K-fold Cross Validation. In ESANN.

[61] Al-Zewairi, M., Almajali, S., & Awajan, A. (2017, October). Experimental Evaluation of a Multi-layer Feed-Forward Artificial Neural Network Classifier for Network Intrusion Detection System. In New Trends in Computing Sciences (ICTCS), 2017 International Conference on (pp. 167-172). IEEE.

[62] Primartha, R., & Tama, B. A. (2017, November). Anomaly detection using random forest: A performance revisited. In Data and Software Engineering (ICoDSE), 2017 International Conference on (pp. 1-6). IEEE.

[63] Gopalani, S., & Arora, R. (2015). Comparing apache spark and map reduce with performance analysis using k-means. International journal of computer applications, 113(1).

[64] Dahiya, P., & Srivastava, D. K. (2018). Network Intrusion Detection in Big Dataset Using Spark. Procedia Computer Science, 132, 253-262.

[65] Belouch, M., El Hadaj, S., & Idhammad, M. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Computer Science, 127, 1-6.

[66] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on (pp. 1-6). IEEE.

[67] KDD Cup 1999 avalable on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[68] Dhanabal, L., & Shantharajah, S. p. (2015). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. International Journal of Advanced Research in Computer and Communication Engineering, 4(6), 446-452.

[69] NLS_KDD avalable on: https://www.unb.ca/cic/datasets/nsl.html.

[70] Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1-3), 18-31.

[71] Moustafa, N., & Slay, J. (2015, November). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Military Communications and Information Systems Conference (MilCIS), 2015 (pp. 1-6). IEEE.

[72] http://www.cybersecurity.unsw.adfa.edu.au/ADFA%20NB15%20Datasets/

[73] https://www.ixiacom.com/products/perfectstorm

[74] https://qosient.com/argus/index.shtml

[75] https://www.bro.org/

[76] http://www.unb.ca/cic/datasets/ids-2017.html.

[77] Sharafaldin, I., Gharib, A., Lashkari, A. H., & Ghorbani, A. A. (2018). Towards a reliable intrusion detection benchmark dataset. Software Networking, 2018(1), 177-200.

[78] Lashkari, A. H., Draper-Gil, G., Mamun, M. S. I., & Ghorbani, A. A. (2017). Characterization of Tor Traffic using Time based Features. In ICISSP (pp. 253-262).

[79] Belouch, M., El Hadaj, S., & Idhammad, M. (2017). A two-stage classifier approach using reptree algorithm for network intrusion detection. International Journal of Advanced Computer Science and Applications (ijacsa), 8(6), 389-394.

[80] Gharaee, H., & Hosseinvand, H. (2016, September). A new feature selection IDS based on genetic algorithm and SVM. In Telecommunications (IST), 2016 8th International Symposium on (pp. 139-144). IEEE.

[81] Vijayanand, R., Devaraj, D., & Kannapiran, B. (2018). Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. Computers & Security, 77, 304-314.

[82] Resende, p. A. A., & Drummond, A. C. (2018). Adaptive anomaly-based intrusion detection system using genetic algorithm and profiling. Security and Privacy, 1(4), e36.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Faker, Osama
Nationality: Libya
Date and Place of Birth: 4 December 1979, Sabha
Marital Status: Married
E-mail: osamafakre@gmail.com

EDUCATION

Degree	Institution	Year of Graduation
BA	Sebha University, Sabha – Faculty of Software Engineering	2005
High School	Oqba bin Nafi High School, Sabha	1997

WORK EXPERIENCE

Year	Institution	Position
2006-2009	Higher Institute for Raising the Efficiency of Trainers	Employee
2009-now	General Electric Company	Employee

FOREIGN LANGUAGES

English

AREAS OF INTEREST

Literature, Sport, History