DEEP LEARNING ALGORITHM FOR THE DETECTION OF BOTNET ATTACK IN CLOUD ENVIRONMENT

BY

BAHAGO, Habiba Abubakar
MTech/SICT/2017/6806

A THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, NIGERIA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF MASTER OF TECHNOLOGY IN CYBER SECURITY SCIENCE

SEPTEMBER, 2021

ABSTRACT

The upsurge in the use of computer, internet, and cloud-based resources and vectors in everyday dealings for a lot of businesses or general communication due to its cost-effectiveness and efficiency has made cloud services vulnerable to attacks including DDoS. Botnets also called Bots is the number of Internet-connected devices, which is running one or more bots each. Botnets can be used to perform distributed denial-of-service attack, steal data, send spam, and allows attackers to access devices and their connections. In this study, a proposed Deep Neural Network learning algorithm for cloud botnet attack is presented which comprises of 5 layers architecture. In which the first layer is the input, the second, third and fourth layers are the hidden layer, while the fifth layer represents the output later. In the input layer 42 neurons is composed which equate to the number of features found in the adopted dataset. The processing of the input data for the model building takes place in hidden layer through weights manipulation, the input data was transformed based on the non-linear function known as activation function. The activation function adopted for this research are rectified linear unit (RELU) for the hidden layers and sigmoid function for the output layer. The performance of the algorithm was measured in terms of accuracy, sensitivity, false positive rate, false alarm rate and detection rate respectively. The achievement recorded from the proposed DNN learning algorithm model indicates that an accuracy of 99.65% was achieved which reflects to the high strength of the proposed model in distinguishing a cloud botnet attack from a normal internet traffic flows, in the same manner sensitivity score of 99.68% was achieved from the proposed model while 3.5% and 0.4% was achieved for false alarm rate (FAR) and False positive rate (FPR) respectively.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IRC	Internet Relay Chart
C&C	Command and Control
DL	Deep Learning
P2P	Peer to Peer
HTTP	Hypertext Transfer Protocol
SVM	Support Vector Machine
GA	Genetic Algorithm
AFSA	Artificial Fish Swarm Algorithm
TCP	Transport Communication Protocol
KNN	K- Nearest Neighbour
LSTM	Long Short-Term Memory
DDOs	Distributed Denial of Service
USLM	Unsupervised Machine Learning
DNN	Deep Neural Network
FAR	False Alarm Rate
FPR	False Positive Rate
ReLU	Rectified linear Unit
NN	Neural network
TPR	True Positive Rate
RAT	Remote Access Trojan

CHAPTER ONE

1.0 INTRODUCTION

1.1 Background of the Study

Cloud computing schemes have now supplied a broad range of computing services power that allow companies to offer space on its physical mechanisms for an hourly cost in order to maintain a consistent turnover(Deka *et al.,* 2017). Cloud computing is a technology that managesflow of information as well as special programsthrough the service of internet connectivity and central cloud infrastructure.Cloud servicesuser and companies can leverage cloud computing programs without installing them and working their documents from any computing devices with internet connectivity (Deka *et al*., 2017). Many individuals have associated cloud computing with utility computing or grid computing, and many businesses that claim to provide cloud computing actually provide utility computing services.(Somani *et al.,* 2017). In a grid computing systemZissis and Lekkas (2012),networked gadgets can access and utilize the services of all other networked infrastructures, however cloud computing technologies normally only apply to the backends. Utility computing is a business model where one company pays another company for access to computer applications or data storage. In this way, utility computing is relatively straightforward, cloud computing, on the other hand, becomes less direct,while all of the services are still being hired, the enterprise has little information on where the services come from. Users continue to pay for what they consume, but the company offering the services use far more advance infrastructure and software technology, which frequently involves complex networks that offers several jobs at a time.As a result, cloud computing is more of advance technology because it is not reliant on a single platform for its resource power. Cloud computing, distribute the task functionality associated with it, to

facilitated fast, effective and efficient way of carrying out schedules task, with less diagnosing complexity in general(Zissis and Lekkas, 2012).

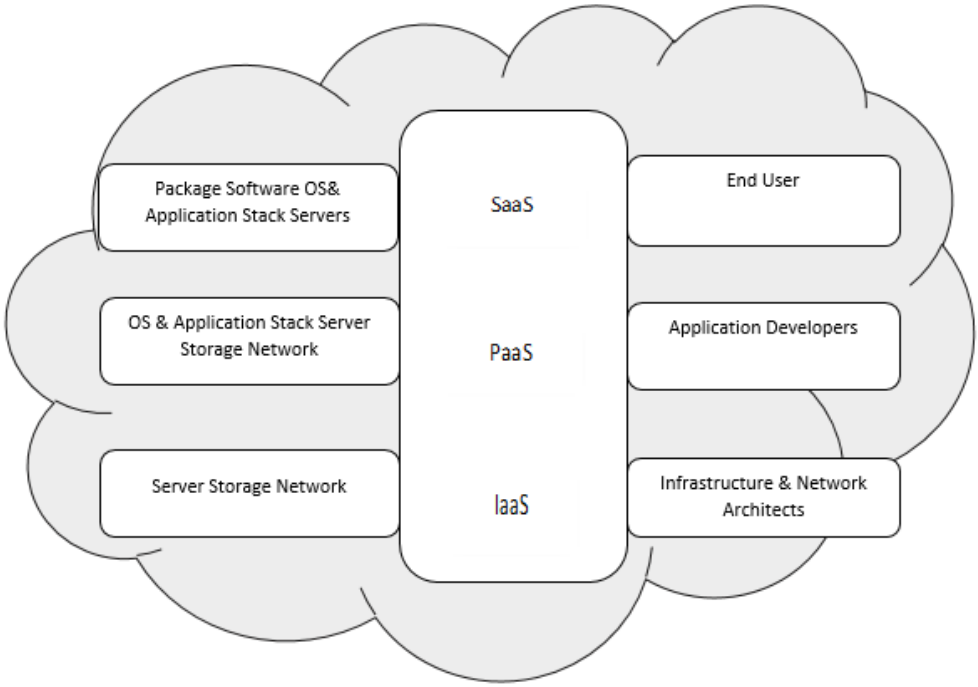

Figure1.1 Cloud Service Model(Zissis and Lekkas, 2012)

Software-as-a-service (SaaS): The programs are hosted by a cloud service provider and made available to consumers via the internet in this model.

Platform-as-a-service: This is a programming environment for programmers. It is created for programmers to use in order to design, execute, test, and manage programs.

Infrastructure-as-a-service (IaaS): It's a method of offering cloud computing infrastructure, such as servers, storage, networking, and operating systems. Rather than purchasing a server or software, IaaS allows you to buy all of the resources you need.

Because of its numerous advantages over other standard Machine Learning (ML) techniques, Deep Learning (DL) model have demonstrated a vital position in handling complex issues. DL is a collection of multi-layered machine learning algorithms that are capable of learning high-level abstractions from complex large-scale data.Many non-linear

hidden layers are used in DL algorithms to learn feature representations, making feature engineering automatic. DL is now widely employed in practically all applications. Itis term as new variant of the classical Multi-Layer Perceptron(MLP). DL model is targeted at producing enhancedand optimal features from the raw data input to help in terms of well generalized classification.

1.2 Statement of the Problem

Botnet attack has been a wide threat to connected computed device, and the literatures reviewed that machine learning algorithm have been employed in classification of botnet while performance in terms of accuracy and false positive rate serve as a major challengein respect to enhance performance as recorded byTuan *et al.*(2019) and Azzaoui *et al*. (2021) with an accuracy of 0.9808 and 0.9963 respectivelyand a high false positive rate of 0.9188 and 0.110 respectively.

1.3 Aim and Objectives of the Study

The aim of this research is to design a deep neural network learning algorithm for cloud-botnet detection. The objectives to achieve this aim are to:

i. design a deep learning algorithm-based framework for the detection of botnet attack in cloud environment.
ii. formulate a deep learning algorithm based mathematical model for the detection of botnet attack in cloud environment.
iii. validate the proposed model in (ii) through experimentation.
iv. evaluate the performance of deep neural network model for the detection of botnet attack using relevant performance metrics.

1.4Significance of the Study

One of the most important aspects of the proposed research is that it will help to better distinguish cloud botnets.

Botnets cost customers, service providers, network operators, and society as a whole economic and social expenses. Botnets have the potential to impair the internet's overall economic and social benefits if they are not effectively mitigated.

Botnets can spread across distance and geography, with infected computers and botnet herders operating in different countries and locations. As a result, botnet detection, mitigation, and law enforcement require a collaborative approach. This research provides insight into a classification algorithm with optimal performance that can be used to mitigate botnet attacks in cloud computing.

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1Botnets Preamble

A botnet is a group of Internet-connected machines with each containing more than a bot infected device. Botnets have the capability of performing a distributed denial of service attack, data theft, email spam, as well as give threat actors avenue to a device's network(Antonakakis *et al*., 2017). Over the previous decade, the botnet has evolved into a highly dangerous phenomena that has demonstrated its negative impact on network communities. Researchers, law enforcement agencies, organizations, and individuals have begun to develop strategies to counteract this dangerous menace.

The botnet has grown in prominence as a means of disseminating a variety of Internet threats, such as spamming, Distributed Denial of Service (DDoS) threats, and nefarious activity. A botnet is a collection of compromised devices (sometimes known as "bots") that work together in order to spread nefarious code across the connected technology without the need for human interaction.This procedure is managed by a centralized entity known as command and control (C&C), also known as a "botmaster"(Alomari*et al*., 2012).As a result, the concept of the Command and Control (C&C) modeloffers an expanded number of bot adversaries and coordinate these adversaries in order to carry out extensive destructive operations. The presence of C & C distinguishes a botnet from other types of network attacks. Furthermore, the compromised machines (bots) get instructions from C & C and execute them accordingly.The instructions/commands range from launching an Internet-based worm or spam attack to interfering with a genuine user request. Bots are computer machines that have been infected with malicious programs (Carlin *et al.*,2015).Deep learning (DL), a new model in data science as well machine learning, have

been deployed in industries andresearch environment for different applications such as computer technology vision, based inleveragedstrength(Bengio, 2009: Lv*et al.*, 2015).In terms of feature learning, a multilayer deep learning architecture is superior, more crucially, deep model architectures overcome the learning difficulties by layer wise pretraining, which entails pretraining many layers of feature detectors from the lowest to the highest level in order to build the final classification models(Lv et al., 2015).This envisages a model devised for malware detection based on deep learning.

A collaborative approach for detecting RAT-bots was developed in this study. Machine learning technologies, including Host, were used to construct a two-phase decision-making process for detecting infected RAT hosts.

2.2Botnet Lifecycle

A botnet lifecycle (Feily, 2009)comprise of initial stage, command and control (C&C), attack, and post-attack, are the four phases the attack. In the first attack phase, botmaster exploits a compromised system's weakness and exploit the victim device through injecting a maliciousscript. In the third attack phase, the hacked computer is made to run malicious code.Second phase, the bot will establish a link to the command-and-control channel before confidently activating the botnet. After establishing a connection with the command and control (C&C) channel, the bot will seek the botmaster for permission to undertake the malicious operations in the attack phase.Updates of the scripts loaded in the initial phase can generally be performed in the post-attack phase to protect against detection.

2.2.1 Categories of botnet based on command-and-control channel

Botnets are categorized based on four groups with regards to command and control (C&C) channel model. Figure 2.1 present a block diagram of the four botnet categories.The Internet Relay Chat (IRC) botnet, Peer-to-Peer (P2P) botnet, Hyper Text Transfer Protocol (HTTP) botnet, and the hybrid botnet, which combines all sorts of botnet structures, are the four forms of botnets. Botmasters have recently used SMS and blue-tooth as command and control (C&C) channels to execute harmful actions on highly complex mobile phones such as Smartphones.Such botnets usually classified as mobile botnets; however, botnets can still be developed using cloud techniques, which are known as bot cloud botnets(Feily, 2009).

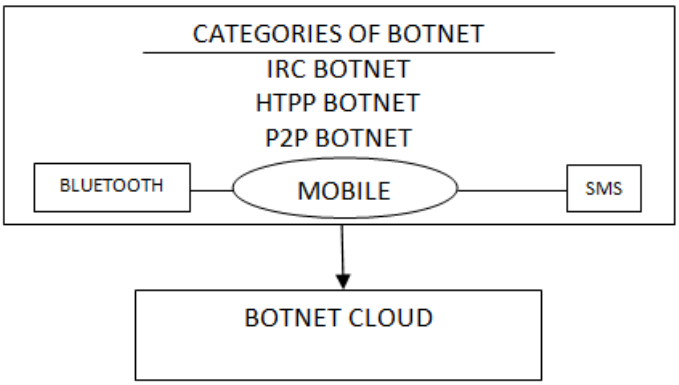

Figure2.1 Categories of Botnet(Feily, 2009)

A. Internet relay chat botnet

An Internet Relay Chat (IRC) botnet is a collection of malware-infected devices that may be remotely controlled via an IRC channel. It entails a botnet operator controlling IRC bots via the previously set IRC server and channel. IRC can also be used to detect and prevent malicious IRC Channel interventions, as well as to

prevent researchers from automating particular tasks. All messages will be lost if the IRC server is shut off.

A botmaster's command and control of botnets for remote process execution is depicted in Figure 2.2. Botnets are installed on infected workstations via a variety of remote code installation methods. The botmaster will conceal his or her identify in order to prevent investigators and law enforcement from detecting their IP address(Feily, 2009).

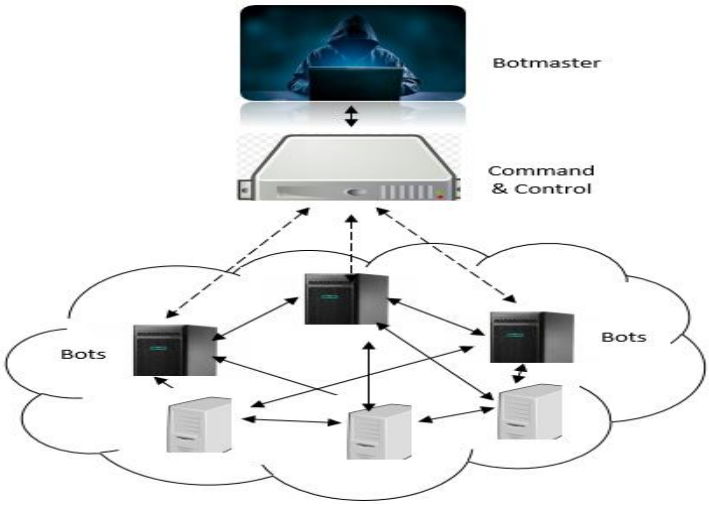

Figure2.2 IRC Botnet(Feily, 2009)

B Peer topeer botnets

A peer-to-peer botnet is a decentralized collection of software that infects devices without the owners' consent. Botnets that run peer-to-peer (P2P) do so without the use of a command and control (C&C) server. Implementing a peer-to-peer (P2P) botnet is also challenging and sophisticated. Botnets that operate on a peer-to-peer

(P2P) network are more difficult to track, shut down, observe, and take control over. Figure2.3illustrates ways systems can be hacked without the users being knowing.

Figure 2.3 P2P Botnet

C. Hypertext transfer protocol (http) botnets

Hypertext transfer protocol (HTTP) protocol is used to distribute commands on web servers instead of staying connected. Hypertext Transfer Protocol (HTTP) bot gets updates and new commands from the server. The botmaster hides activities and avoids current detection methods like firewall. The detection of Hypertext Transfer Protocol (HTTP) botnets is not good whenever the botmaster uses the legitimate website to create their command and controls(Feily, 2009).

D. Mobile botnet

A mobile botnet is a form of botnet that tries to obtain access to and control of mobile devices such as Smartphones by attempting to gain access to the device and all of its contents.A mobile botnet is a collection of infected cellphones that share the similar command and control technology and are used to carry out harmful attacks by a bot master. Botmaster can also operate the bots via both Bluetooth and SMS as a command and control (C&C) channel(Feily, 2009).

E. Cloud botnet

Botmasters uses cloud services to make botnet François *et al*. (2012) serving as a tremendous challenge experienced by the cloud service providers due tocomplication associated with detecting botnet in a highly dynamic and diversetechnology,botnets are also called bot clouds.

A novel approach was implemented to enable the detection of botnet on the record obtained from NetFlow protocol as well as it is clustering method, a dendrogram was developed that reveals relationships that exist between bots based on the clusters generated from events, alerts, and activities of network threats as obtained from NetFlow. The research achieved a low false positive rate in terms of botnet detection through the merging of the modified cluster in respect to some rules, and other information as it relates to an identified node that was infected with botnet, furthermore, GNS3 was used as a simulation environment(Amini *et al.,* 2014).

The study as contain in Syahirah *et al*. (2013) focus basically in reviewing and articulating techniques employing in botnet detection of which anomaly based method was said to encompass DNS detection, Data mining focused on detection, host-based detection, and network-based detection method were more prominent in most of the researches based on the effectiveness in botnet detection, prevention as well as mitigation in particular unknown botnet, while it was ascertained that signature-based method was less effective in terms of botnet detection.

The following machine learning algorithms was employed in this research to detect and classify Distributed Denial of Service (DDoS) attack; k-Nearest Neighbor (KNN), Support Vector Machine (SVM) Random Forest (RF) and Naïve Base, RF algorithm was implemented at the filtering stage of DDoSattack, while KNN was used for classification of

feature in the proposed methodology which entails training and testing phases, meanwhile, the study focus majorly on DDoS attack based on User Datagram Protocol (UDP) flooding (Bandara *et al*., 2016).

In the survey Silva, *et al.*(2013) as it relates to botnet,, it was stated that botnet stand as a major security challenge to Internet, as 80% of spam attack is as a result of compromise network node by botnet, botnet was further describe as having a destructive capability against connected computing nodes, while mitigating of such challenge can be achieve through the development of strong detection methods such as honeynet, intrusion detection system (IDS) which encompasses signature and anomaly-based detection method, of which anomaly-based detection method is regarded as most consider technique by many researchers due to its optimal performance.

A proposed framework that involves passive monitoring of network traffic was opined in this study; the model focuses on a P2P Botnet that leads to the multiplication of bots in the same network. The proposed botnet detection model is made up the following; Filtering, Application Classifier, Traffic Monitoring, Malicious Activity Detection, Analyzer, Monitoring and Clustering, and Flaws Analyzer, it was asserted that the botnet is an outstanding factor of the proposed framework (Zeidanloo and Manaf, 2010).

Support Vector Machine (SVM) and Artificial Fish Swarm Algorithm (AFSA) was used to achieve superior performance against Genetic Algorithm (GA) in relation to botnet detection with regards to botnet feature classification as it relates to the experiment performed on dataset obtained from Internet Relay Chat (IRC) in the proposed botnet feature characterization method in this research. Furthermore, a LAN environment composing of multiple nodes infected with botnet malware serve as a simulating platform

in evaluating the proposed model, however, the proposed model has weakness in detecting dynamic nature employ by some botnet attack (Lin et al., 2014).

The option of 10 folds was due to results obtained from broad tests on various datasets, with different learning procedures, that have proven that 10 is about the correct number of folds to get the best gauge of error (Abdulhamid*et al*., 2018; Mahardhika*et al.*,2017). Finally, the averages of the 10 error estimates are taken to give an overall error estimate (Abdulhamid *et al*., 2018; Syahirah *et al.,* 2013). A proposed network flow employing traffic logs approach on the dataset was carried out in this study by creating a data model using a pair of source IP (Internet Protocol), destination IP, and source port, destination port over time to extract new characteristics. The ISCX Botnet Dataset was subjected to Decision Tree, Naive Bayes, Rule Induction, and K-Nearest Neighbor analysis. All algorithms used achieved high-rate precision, Recall, F-measure and accuracy. Rule induction have best performance with Precision98.70%, F-Measure99.40%, Recall98.80%, and Accuracy of 98.80% but have low speed (42.430ms). The result of K-Nearest Neighbor is moststablethan all algorithms that achieve precision, Recall, F measure and accuracy to 98.10% and high speed (50 ms). Decision Tree achieves lower performance than rule induction, but better than K-Nearest Neighbor and Naive Bayes. From the experimental result and analysis above it prove that the supervised ISCX botnet dataset can be implemented using machine learning method and processed to another algorithm(Mahardhika*et al.,* 2017).

In this paper machine learning classifiers was implemented to detect Hypertext Transfer Protocol(HTTP) botnets. Network traffic dataset was employed in the research and was extracted based on TCP packet feature. The proposed methodology was evaluated based on false positive rate, true positive rate and accuracy on five different Hypertext Transfer

Protocol(HTTP) botnets. The classifiers used in the experiment are as follows K- nearest neighbor, Random-forest, Naïve Bayes and Decision tree respectively, In the experiment it shows that KNN emerges as the best classifier by achieving the average accuracy of 92.3% and 95.4% of the true positive rate respectively(Fadhlee *et al*., 2018).

In this research work a new approach for detecting cloud-based botnet was employed in which the approach is used in determining the randomness of the communications between the command-and-control server and the bots. The research work was evaluated using a cloud-based botnet with in a closed networking environment. The proposed approach was based on entropy and was used for the detection of Hypertext Transfer Protocol (HTPP) botnets(Lu*et al.,* 2017).

The study proposed in this paper is to model and validate a solution for detecting malicious activities infected with botnet by virtual host in a public cloud based context(Cogranne*et al.*,2017).

In this study, deep learning techniques were applied by recommending BOTshark-SA that uses auto encoders and BOTshark-CNN that uses CNN in the detection of malicious botnet traffic. The results show that TPR 0.91% with FPR 0.13%was achieved respectively and auto encoders performs (Homayoun et al., 2018).

In this work, machine learning algorithm to detect botnet DDOS attack was analyzed. The algorithms used were Support Vector Machine, Artificial Neural Network, Naïve Bayes, Unsupervised machine learning(K-means-means), the analysis was conducted on the dataset UNBS-NB15 and KDD99.USML has been shown to be the best in distinguishing

between botnet and normal network traffic in terms of accuracy, false alarm rate, MCC, AUC respectively(Tuan*et al*., 2019).

In this study a botnet detection approach based on DNS rules that can increase the accuracy of botnet detection based on DNS traffic was employed. The study is focused on DNS query and response behaviors. The results also shows that the proposed solution detected botnet attacks with 99.35% accuracy and 0.25% false positive rate respectively(Alieyan*et al.,* 2019).

In this research work, a deep neural network based on the contextual long short term-memory (LSTM)architecture that uses both content and metadata to detect tweet-level bot was suggested. The result also shows that from a single tweet the proposed architecture achieved a high accuracy of 96% when distinguishing bots from humans, the same architecture was conducted on account- level to detect botnet and 99%accuracy was achieved respectively(Kudugunta and Ferrara, 2018).

In this study, a new approach to botnet detection that employs deep learning on TCP/UDP/IP-packet flows were proposed. In the result it shows that 99,7% accuracy was achieved in the classification of P2P botnet traffic(Van *et al.*, 2018)

In Mazini*et al.* (2019), the authors of the paper proposed a hybrid method for abnormal network-based IDS detection. The approach combines Artificial Bee Colony (ABC) and Adaboost algorithm with the aim of obtaining high detection value and low false positive value. The result of their approach was evaluated employing NSL-KDD and ISCXIDS2012 datasets. The paper got an accuracy of 99.61% and a false positive rate of 0.01.

Shenfield*et al.* (2018), proposed a new approach for detecting malicious traffic using artificial neural network. The proposed approach was experimented using benign network traffic like dynamic link library files, images, word processing documents, music files. An average accuracy of 98% was obtained. The system was experimented on offline data. Future work would be to integrate the approach into IDS that is capable of detecting intrusion online and to also try it in a life scenarionetwork environment. Applying intelligent method to intrusion detection mentioned here to other areas where network security challenges is another proposed future work, areas like SQL injection attacks on web applications and cross-site scripting attacks.

Nezhad*et al.* (2016)two metrics are used to detect DoS and DDoS attacks: the quantity of packets and the number of source IP addresses. They created a time series based on the number of packets principle and normalized it using the Box-Cox transformation.The researchers employed an ARIMA model to forecast the volume of traffic or packets that would be generated in the next minute. They then computed the greatest exponent to look at the chaotic behavior of prediction error time series. The local exponent is also used to determine whether mistakes are chaotic or non-chaotic.They finally proposed set of rules based on repeatability of chaotic behavior and enormous growth in the ratio of number of packets to number of source address during attack time to classify normal and attack traffics. The result gotten from their simulation showed that algorithm can accurately classify 99.5% of traffic state. The future would be to try and differentiate a busty traffic (traffic whose rate are slightly different from normal traffic but are not attack) form attack traffic.

Khundrakpamand Tanmay (2020) proposed multilayer perception with genetic algorithm to detect DDoS attack at the seventh layer of the OSI model that is the application layer. Four

features were considered from traffics entering the network or computer that exhibit important qualities in their qualities. The second parameter is the number of IP address that enters a network within a little specified time. The third parameter is the constant mapping function. The fourth parameter is the fixed frame length. When there is a change in these features, attack will be detected.

Experiment result reveals that the technique gave 98.03% accuracy in detecting attack at the seventh layer (application layer) with negative positive value of 2.21%. The future work is to work on improving the detection accuracy.

A deep neural network that uses feed forward back propagation architecture and the use of hidden layers was employed in this research work. The proposed research work was able to detect some level of packet flows features with an accuracy of 98% respectively(Asad *et al.*, 2019).

In this study, ANN technique was proposed, more also, the use of deep learning for the detection of botnet attack which achieves an accuracy of 99.6% compared to other machine learning algorithm such as SVM,NB respectively(Ali*et al.,* 2020).

A deep learning-based model to secure a fog network from distributed denial of service(DDOS)attack has been implemented in this research work. Application of the SDN technology was used to monitor the fog network. In addition, a real cloud system with an open source cloud platform was used to setup the fog computing environment by achieving 98.88% accuracy.(Priyadarshini and Barik, 2019)

In this study, a detection system of HTTP DDOS attacks in cloud computing environment based on information theoretic entropy and random forest ensemble learning algorithm was employed. more also, the experiment was performed using five machine learning

algorithms on CIDDS-001 public dataset achieving 99.5% accuracy and 0.04% false positive rate respectively.(Idhammad*et al*., 2018)

An approach for detecting DDOS attacks based on a feature selection is proposed in this study. The proposed study achieved a better accuracy of 98.3% compared to other features. More also, four classification algorithm was employed whereby MLP was selected due to better performance and also a less computation in terms of time with a lower RMSE value was used(Singh and De, 2017).

In this research work, a machine-based learning system was proposed to identify compromised hosts and networks infected with the RAT-bots. The mechanism consists of two agents with the host agent responsible for monitoring the running host's device activity and raising an alert while network agent tracks network traffic to eliminate any suspicious patterns. Furthermore, the experimental results show that the suggested solution achieves an accuracy of 98.83% with a false positive rate of 1.45% respectively(Awad and Sayed, 2019).

In this paper, a novel botnet detection approach based on deep learning was proposed and tested. The performance of deep learning architecture was investigated by tuning the numbers of hidden layers and neurons. Botnet detection was evaluated using two datasets that are publicly accessible for benchmarking. The results of the proposed method shows that the performance obtained outperforms other methods with an accuracy of 99% (Pekta, 2018).

In this study, a new Deep Neural Network model based on Intrusion detection system(IDS) model for network traffic classification is introduced. The CICIDS datasets and NSL-KDD

datasets were used in this experiment. More also, the results are analyzed based on different performance metrics whereby an accuracy of 99.43% and 99.63% using both CICIDS and NSL-KDD datasets was achieved respectively(Azzaoui*et al*., 2021).

In this paper, multilayer system for botnet detection that includes filtering module and a classification module for detecting command and control(C&C) was proposed, certain parameters regarding the architecture includes structure independence, protocol independence and as well as the ability to identify botnet using an encapsulated approach. The use of flow-based features to analyze the packet header by aggregating it to a 1s time. The experiment was performed based on different time intervals, more also, the results performs better when 1s time interval was used by achieving 92% of F-scores and a false negative rate of 1.5% respectively(Nur*et al*., 2021).

An experimental study of machine learning methods for botnet DDOS attack detection was conducted in this research. The performance was done based on UNBS-NB15 and KDD99 datasets respectively. Unsupervised learning (USML) has been shown to be the most efficient at distinguishing between botnet and normal network traffic in terms of accuracy, false alarm rate(FAR) , false positive rate(FPR), MCC, sensitivity, and AUC(Tuan*et al.*, 2019).

In this study, a deep learning based methodology for detecting botnets and identifying a minimal botnet attack was presented, on the CTU-13 dataset the proposed model is trained and tested. In addition, when compared to other machine learning algorithms such as SVM, NB and Back propagation the proposed model achieved an accuracy of 99.6%(Ahmed et al., 2020).

In this research work, deep learning environments with fine-tuned parameters were used to find an optimal performance for the detection of android malware. In addition, a thorough examination of some hyper-parameters was conducted. The result shows that the proposed solution detected an android malware with an accuracy of 95% and 93%F1 score respectively(Booz *et al*., 2018).

An embedded deep learning based expert system for obtaining processed rules from a trained deep neural network was introduced in this study. The proposed model was evaluated using two different datasets, where different scenarios will be launched to check the systems robustness and flexibility as well as comparing the performance of the proposed model with a DNN, JRip, and other machine learning algorithms. More also, the results shows that the proposed model achieved an accuracy of 97.5% and 1.8% of FPR respectively(Booz *et al.*, 2019).

2.3 Machine Learning

The Machine learning is one of the multitudes of DDoS attack techniques which is used in detecting anomalies and changes in a network be it cloud or grid network traffic (inbound and outbound). This techniques is about learning and making futuristic predictions Mahardhika *et al*., (2017) based on earlier experiments and observed data Bandara *et al*., (2016).

2.4 Deep Neural Network

Deep neural network (DNN) which is a part of machine learning mimics the functionality of human knowledge reasoning, Deep learning is recognized as a key component of data science, this is based on its strength in statistical operation as well as predictive modelling, furthermore, deep learning have the strong capability of analyzing and interpreting large

volume of data with enhance predictive strength, deep learning have been widely deployed in various fields of study such as power systemAlmalaq and Edwards (2017), wind speed forecasting Chen *et al*. (2018) as well as cyber physical power system(Wei and Mendis, 2016).

Deep neural network is recognized sub of artificial neural network, which deals with multiple hidden layers, composed generally of input, hidden and output layers, with a mathematical relationship that exist between the input and output layers through the corresponding network weight, which make it to learn complex pattern, by the operation linear or non-linear activation function (Nielsen, 2015).

The efficiency of DNN is noted to be greatly dependent on its hyper-parameters, hence, determining the optimal combinations of right hyper-parameters is key to attaining performance excellent in a model Niazazari*et al*. (2020), part of the options of addressing the aforementioned challenge as stated by Niazazari*et al*. (2020) is manual tuning of hyper-parameter such as trial and error to determine best combination of hyper-parameters for optimal performance through a random search in an iterative process (Niazazari*et al.*, 2020).

A representation of neural network training process is depicted in Figure 2.5, which is composed of definition and configuration of parameter, epoch training, batch size training and an optimizer known as ADAM (Lin*et al.*, 2020)

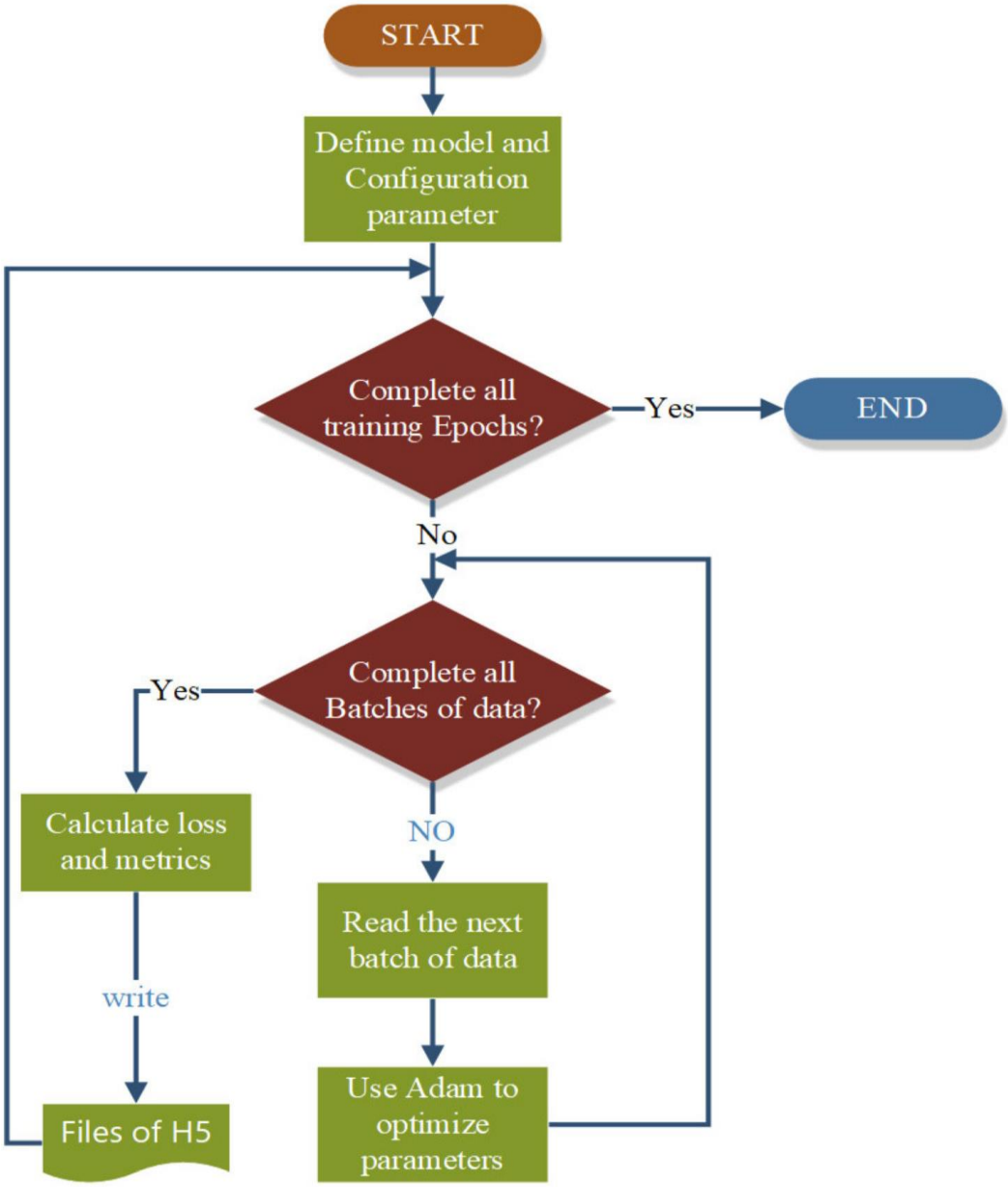

Figure 2.5 The flow chart of neural network training(Lin *et al.*, 2020)

CHAPTER THREE

3.0 RESEACH METHODOLOGY

3.1 Research Design

The frameworkpresented in Figure 3.1 represents the research methodology that wasemployed in designing a deep neural network learning algorithm for cloud botnet detection. This research used the data collected from well-knowndataset repository, implementing a deep neutral network algorithm application for classification and carrying out a performance evaluation.

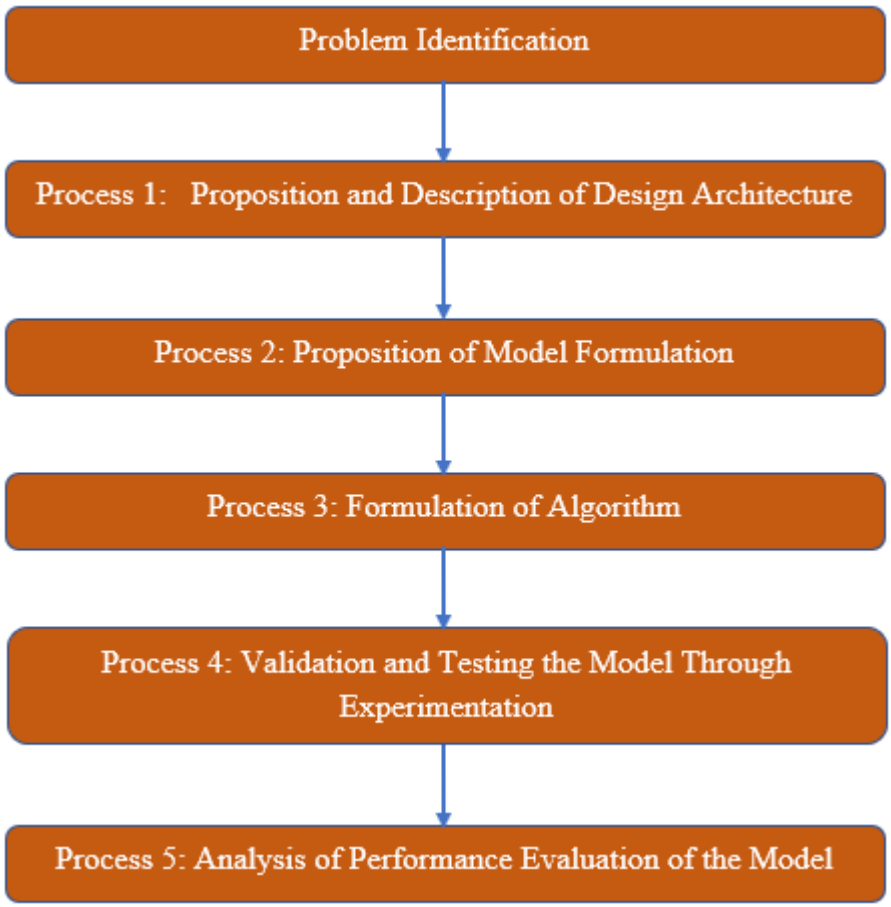

Figure 3.1 Research design for Deep Neural Network Cloud Botnet Classification

3.2Problem Identification

In any research process, the first step is the detection of problems. This thesis is not in any way different from conventional approach to the research process. After a systematic review of the literatures discussed in chapter 2, a void was found that needed to be filled. Consequently, this study experimented the detection of botnet attacks in cloud computing using deep neural network algorithm.

3.3Proposition and Description of Design Framework

The deep learning-based algorithmframework design for botnet attack detection in cloud is presented in Figure 3.2, outlined are the steps involved in thedesign of a deep learning algorithm-based framework for the detection of botnet attack in cloud computing;

step 1: this entails the feature extracted from cloud environment computing devices which is a composition of normal as well as botnet attack instances

step 2: this phase serves as the repository where the dataset is store and accessed for experiment as well as research purposed

step 3: this gives a hint of the composition of the features that are in the cloud botnet dataset that forms the instances, protocol of internet communication, packet size, network connectivity troubleshooting protocol, remote connection protocol

step 4: this step preprocesses the dataset obtained from the preceding step employing techniques such as dataset encoding and normalization in other to achieve an enhance learning performance in the detection of botnet attack in cloud based deep learning algorithm

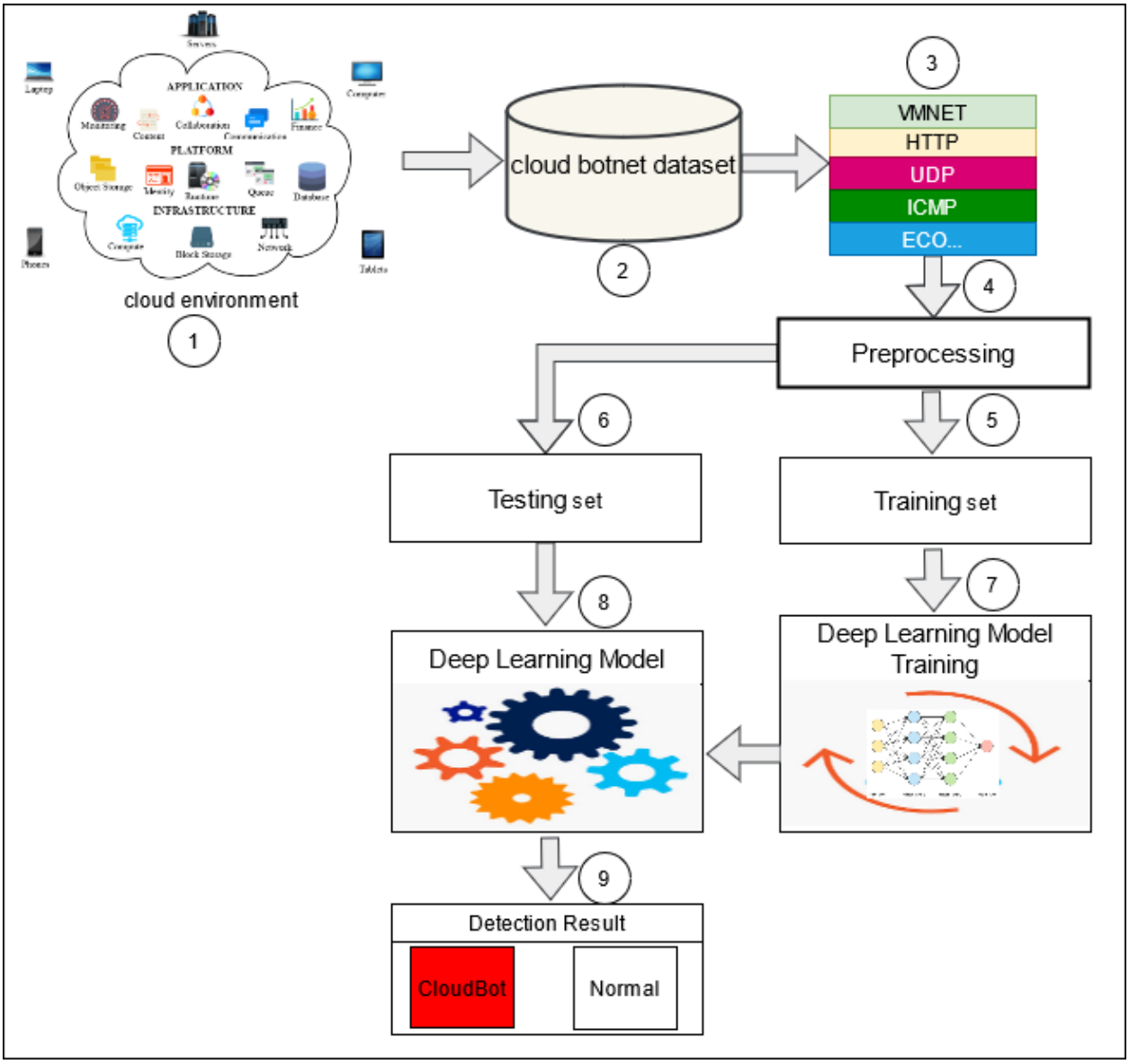

Figure 3.2 Proposed deep learning algorithm-based framework for detection of botnet attack in cloud environment

step 5: this used a defined percentage of the preprocessed cloud botnet dataset from the preceding step to serve as a training set that will be feed into the next step in order to train the candidate model

step 6: this step serves as the testing set, which is also obtained from step 4, the defined percentage is used for the final evaluation of the trained model, this is to determined effectiveness of the proposed deep learning model for botnet attack detection in cloud

step 7: the training set defined in step 5, is used here to build the proposed deep learning model for the detection of botnet attack in cloud

step 8: the final model or rather the trained model from the preceding step is evaluated with the test set defined in step 6

step 9: this present the output generated from the preceding step which is the final model for the detection of botnet attack in cloud.

3.4 Proposed ModelFormulation

The proposed DNN mathematical model formula for cloud botnet attack detection is presented as follow, the equations for the input and output matrix can be presented as:

$$Input = [x_1 x_2 x_3 \quad \cdots \quad x_{42}] \tag{3.1}$$

Where x_i stand for the cloud botnet input features,

$$output = [\bar{y}] \tag{3.2}$$

Where $[\bar{y}]$ is the binary output of normal or cloud botnet attack detection

The equations (3.1) and (3.2) can be combined to form a linear equation (3.3)

$$[\bar{y}] = [weights][x_1 x_2 x_3 \quad \cdots \quad x_{42}] \tag{3.3}$$

$$weights = [w_{ij}][v_{ij}][u_{ij}][t_{ij}] \tag{3.4}$$

Where w_{ij} stand for weights between the input and hidden layers one neurons (l^1)

v_{ij}stand for weights between hidden layers one and two (l^1) and (l^2) neurons

u_{ij}stand for weights between hidden layers two and three (l^2) and (l^3) neurons

t_{ij}stand for weights between hidden layer three and output (l^3) and($\bar{y}$) neurons

The equations (3.5), (3.6), (3.7) and (3.8) stand for the expanded weights equations,

$$w_{ij} = \begin{bmatrix} w_{1\,1}w_{1\,2}w_{1\,3}w_{1\,4} & \cdots & w_{1\,29} \\ w_{2\,1}w_{2\,2}w_{2\,3}w_{2\,4} & \cdots & w_{2\,29} \\ w_{3\,1}w_{3\,2}w_{3\,3}w_{3\,4} & \cdots & w_{3\,29} \\ w_{4\,1}w_{4\,2}w_{4\,3}w_{4\,4} & \cdots & w_{4\,29} \\ \vdots \quad \vdots \quad \vdots & \vdots & \ddots \quad \vdots \\ w_{42\,1}w_{42\,2}w_{42\,3}w_{42\,4} & \cdots & w_{42\,29} \end{bmatrix} \tag{3.5}$$

$$v_{ij} = \begin{bmatrix} v_{1\,1}v_{1\,2}v_{1\,3}v_{1\,4} & \cdots & v_{1\,29} \\ v_{2\,1}v_{2\,2}v_{2\,3}v_{2\,4} & \cdots & v_{2\,29} \\ v_{3\,1}v_{3\,2}v_{3\,3}v_{3\,4} & \cdots & v_{3\,29} \\ v_{4\,1}v_{4\,2}v_{4\,3}v_{4\,4} & \cdots & v_{4\,29} \\ \vdots \quad \vdots \quad \vdots & \vdots & \ddots \quad \vdots \\ v_{29\,1}v_{29\,2}v_{29\,3}v_{29\,4} & \cdots & v_{29\,29} \end{bmatrix} \tag{3.6}$$

$$u_{ij} = \begin{bmatrix} u_{1\,1}u_{1\,2}u_{1\,3}u_{1\,4} & \cdots & u_{1\,29} \\ u_{2\,1}u_{2\,2}u_{2\,3}u_{2\,4} & \cdots & u_{2\,29} \\ u_{3\,1}u_{3\,2}u_{3\,3}u_{3\,4} & \cdots & u_{3\,29} \\ u_{4\,1}u_{4\,2}u_{4\,3}u_{4\,4} & \cdots & u_{4\,29} \\ \vdots \quad \vdots \quad \vdots & \ddots & \vdots \\ u_{29\,1}u_{29\,2}u_{29\,3}u_{29\,4} & \cdots & u_{29\,29} \end{bmatrix} \tag{3.7}$$

$$t_{ij} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \\ \vdots \\ t_{29} \end{bmatrix} \tag{3.8}$$

Equations (3.5) by (3.6) give equation (3.9)

$$[w_{ij}][v_{ij}] = \begin{bmatrix} w_{1\,1}w_{1\,2}w_{1\,3}w_{1\,4} & \cdots & w_{1\,29} \\ w_{2\,1}w_{2\,2}w_{2\,3}w_{2\,4} & \cdots & w_{2\,29} \\ w_{3\,1}w_{3\,2}w_{3\,3}w_{3\,4} & \cdots & w_{3\,29} \\ w_{4\,1}w_{4\,2}w_{4\,3}w_{4\,4} & \cdots & w_{4\,29} \\ \vdots \quad \vdots \quad \vdots & \ddots & \vdots \\ w_{42\,1}w_{42\,2}w_{42\,3}w_{42\,4} & \cdots & w_{42\,29} \end{bmatrix} \begin{bmatrix} v_{1\,1}v_{1\,2}v_{1\,3}v_{1\,4} & \cdots & v_{1\,29} \\ v_{2\,1}v_{2\,2}v_{2\,3}v_{2\,4} & \cdots & v_{2\,29} \\ v_{3\,1}v_{3\,2}v_{3\,3}v_{3\,4} & \cdots & v_{3\,29} \\ v_{4\,1}v_{4\,2}v_{4\,3}v_{4\,4} & \cdots & v_{4\,29} \\ \vdots \quad \vdots \quad \vdots & \ddots & \vdots \\ v_{29\,1}v_{29\,2}v_{29\,3}v_{29\,4} & \cdots & v_{29\,29} \end{bmatrix} \tag{3.9}$$

$$[w_{ij}][v_{ij}] =$$

$$\begin{bmatrix} w_{1\,1}v_{1\,1} + w_{1\,2}v_{2\,1} + \cdots + w_{1\,29}v_{29\,1} & w_{1\,1}v_{1\,2} + w_{1\,2}v_{2\,2} + \cdots + w_{1\,29}v_{29\,2} & \cdots & w_{1\,1}v_{1\,29} + w_{1\,2}v_{2\,29} + \cdots + w_{1\,29}v_{29\,29} \\ w_{2\,1}v_{1\,1} + w_{2\,2}v_{2\,1} + \cdots + w_{2\,29}v_{29\,1} & w_{2\,1}v_{1\,2} + w_{2\,2}v_{2\,2} + \cdots + w_{2\,29}v_{29\,2} & \cdots & w_{2\,1}v_{1\,29} + w_{2\,2}v_{2\,29} + \cdots + w_{2\,29}v_{29\,29} \\ w_{3\,1}v_{1\,1} + w_{3\,2}v_{2\,1} + \cdots + w_{3\,29}v_{29\,1} & w_{3\,1}v_{1\,2} + w_{3\,2}v_{2\,2} + \cdots + w_{3\,29}v_{29\,2} & \cdots & w_{3\,1}v_{1\,29} + w_{3\,2}v_{2\,29} + \cdots + w_{3\,29}v_{29\,29} \\ w_{4\,1}v_{1\,1} + w_{4\,2}v_{2\,1} + \cdots + w_{4\,29}v_{29\,1} & w_{4\,1}v_{1\,2} + w_{4\,2}v_{2\,2} + \cdots + w_{4\,29}v_{29\,2} & \cdots & w_{4\,1}v_{1\,29} + w_{4\,2}v_{2\,29} + \cdots + w_{4\,29}v_{29\,29} \\ \vdots & \vdots & \ddots & \vdots \\ w_{42\,1}v_{1\,1} + w_{42\,2}v_{2\,1} + \cdots + w_{421\,29}v_{29\,1} & w_{42\,1}v_{1\,2} + w_{42\,2}v_{2\,2} + \cdots + w_{42\,29}v_{29\,2} & \cdots & w_{42\,1}v_{1\,29} + w_{42\,2}v_{2\,29} + \cdots + w_{42\,29}v_{29\,29} \end{bmatrix}$$

(3.9)

Let equation (3.9) be represented as equation (3.10)

$$[a_{ij}] = \begin{bmatrix} a_{1\,1} & a_{1\,2} & a_{1\,3} & \cdots & a_{1\,29} \\ a_{2\,1} & a_{2\,2} & a_{2\,3} & \cdots & a_{2\,29} \\ a_{3\,1} & a_{3\,2} & a_{3\,3} & \cdots & a_{3\,29} \\ a_{4\,1} & a_{4\,2} & a_{4\,3} & \cdots & a_{4\,29} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{42\,1} & a_{42\,2} & a_{42\,3} & \cdots & a_{42\,29} \end{bmatrix} \quad (3.10)$$

Equation (3.10) and (3.7) gives equation (3.11)

$$[a_{ij}][u_{ij}] = \begin{bmatrix} a_{1\,1} & a_{1\,2} & a_{1\,3} & \cdots & a_{1\,29} \\ a_{2\,1} & a_{2\,2} & a_{2\,3} & \cdots & a_{2\,29} \\ a_{3\,1} & a_{3\,2} & a_{3\,3} & \cdots & a_{3\,29} \\ a_{4\,1} & a_{4\,2} & a_{4\,3} & \cdots & a_{4\,29} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{42\,1} & a_{42\,2} & a_{42\,3} & \cdots & a_{42\,29} \end{bmatrix} \begin{bmatrix} u_{1\,1} & u_{1\,2} & u_{1\,3} & u_{1\,4} & \cdots & u_{1\,29} \\ u_{2\,1} & u_{2\,2} & u_{2\,3} & u_{2\,4} & \cdots & u_{2\,29} \\ u_{3\,1} & u_{3\,2} & u_{3\,3} & u_{3\,4} & \cdots & u_{3\,29} \\ u_{4\,1} & u_{4\,2} & u_{4\,3} & u_{4\,4} & \cdots & u_{4\,29} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{29\,1} & u_{29\,2} & u_{29\,3} & u_{29\,4} & \cdots & u_{29\,29} \end{bmatrix}$$

$$[a_{ij}][u_{ij}] =$$

$$\begin{bmatrix} a_{1\,1}u_{1\,1} + a_{1\,2}u_{2\,1} + \cdots + a_{1\,29}u_{29\,1} & a_{1\,1}u_{1\,2} + a_{1\,2}u_{2\,2} + \cdots + a_{1\,29}u_{29\,2} & \cdots & a_{1\,1}u_{1\,29} + a_{1\,2}u_{2\,29} + \cdots + a_{1\,29}u_{29\,29} \\ a_{2\,1}u_{1\,1} + a_{2\,2}u_{2\,1} + \cdots + a_{2\,29}u_{29\,1} & a_{2\,1}u_{1\,2} + a_{2\,2}u_{2\,2} + \cdots + a_{2\,29}u_{29\,2} & \cdots & a_{2\,1}u_{1\,29} + a_{2\,2}u_{2\,29} + \cdots + a_{2\,29}u_{29\,29} \\ a_{3\,1}u_{1\,1} + a_{3\,2}u_{2\,1} + \cdots + a_{3\,29}u_{29\,1} & a_{3\,1}u_{1\,2} + a_{3\,2}u_{2\,2} + \cdots + a_{3\,29}u_{29\,2} & \cdots & a_{3\,1}u_{1\,29} + a_{3\,2}u_{2\,29} + \cdots + a_{3\,29}u_{29\,29} \\ a_{4\,1}u_{1\,1} + a_{4\,2}u_{2\,1} + \cdots + a_{4\,29}u_{29\,1} & a_{4\,1}u_{1\,2} + a_{4\,2}u_{2\,2} + \cdots + a_{4\,29}u_{29\,2} & \cdots & a_{4\,1}u_{1\,29} + a_{4\,2}u_{2\,29} + \cdots + a_{4\,29}u_{29\,29} \\ \vdots & \vdots & \ddots & \vdots \\ a_{42\,1}u_{1\,1} + a_{42\,2}u_{2\,1} + \cdots + a_{42\,29}u_{29\,1} & a_{42\,1}u_{1\,2} + a_{42\,2}u_{2\,2} + \cdots + a_{42\,29}u_{29\,2} & \cdots & a_{42\,1}u_{1\,29} + a_{42\,2}u_{2\,29} + \cdots + a_{42\,29}u_{29\,29} \end{bmatrix}$$

(3.11)

Let equation (3.11) be represented as equation (3.12)

$$[b_{ij}] = \begin{bmatrix} b_{1\,1} & b_{1\,2} & b_{1\,3} & \cdots & b_{1\,29} \\ b_{2\,1} & b_{2\,2} & b_{2\,3} & \cdots & b_{2\,29} \\ b_{3\,1} & b_{3\,2} & b_{3\,3} & \cdots & b_{3\,29} \\ b_{4\,1} & b_{4\,2} & b_{4\,3} & \cdots & b_{4\,29} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{42\,1} & b_{42\,2} & b_{42\,3} & \cdots & b_{42\,29} \end{bmatrix} \qquad (3.12)$$

Equations (3.12) and (3.8) gives equation (3.13)

$$[b_{ij}][t_{ij}] = \begin{bmatrix} b_{1\,1} & b_{1\,2} & b_{1\,3} & \cdots & b_{1\,29} \\ b_{2\,1} & b_{2\,2} & b_{2\,3} & \cdots & b_{2\,29} \\ b_{3\,1} & b_{3\,2} & b_{3\,3} & \cdots & b_{3\,29} \\ b_{4\,1} & b_{4\,2} & b_{4\,3} & \cdots & b_{4\,29} \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ b_{42\,1} & b_{42\,2} & b_{42\,3} & \cdots & b_{42\,29} \end{bmatrix} \begin{bmatrix} t_{1\,1} \\ t_{2\,1} \\ t_{3\,1} \\ t_{4\,1} \\ \vdots \\ t_{29\,1} \end{bmatrix}$$

$$[b_{ij}][t_{ij}] = \begin{bmatrix} b_{1\,1}t_{1\,1} + b_{1\,2}t_{2\,1} + \cdots + b_{1\,29}t_{29\,1} \\ b_{2\,1}t_{1\,1} + b_{2\,2}t_{2\,1} + \cdots + b_{2\,29}t_{29\,1} \\ b_{3\,1}t_{1\,1} + b_{3\,2}t_{2\,1} + \cdots + b_{3\,29}t_{29\,1} \\ b_{4\,1}t_{1\,1} + b_{4\,2}t_{2\,1} + \cdots + b_{4\,29}t_{29\,1} \\ \iddots \\ b_{42\,1}t_{1\,1} + b_{42\,2}t_{2\,1} + \cdots + b_{42\,29}t_{29\,1} \end{bmatrix} \quad (3.13)$$

Let equation (3.13) be represented as equation (3.14)

$$[c_{ij}] = \begin{bmatrix} c_{1\,1} \\ c_{2\,1} \\ c_{3\,1} \\ c_{4\,1} \\ \vdots \\ c_{42\,1} \end{bmatrix} \tag{3.14}$$

Through the substituting of all these equations into equation (3.3), it gives equation (3.15), thus the formulated mathematical model for deep neural network algorithm for cloud-bot detection is presented in equation (3.15),

$$[c_{ij}][input] = \begin{bmatrix} c_{1\,1} \\ c_{2\,1} \\ c_{3\,1} \\ c_{4\,1} \\ \vdots \\ c_{42\,1} \end{bmatrix} [x_{1\,1} \quad x_{1\,2} \quad x_{1\,3} \quad x_{1\,4} \quad \cdots \quad x_{1\,42}]$$

$$[\bar{y}] = \begin{bmatrix} c_{1\,1}x_{1\,1} & c_{1\,1}x_{1\,2} & c_{1\,1}x_{1\,3} & c_{1\,1}x_{1\,4} & \cdots & c_{1\,1}x_{1\,42} \\ c_{2\,1}x_{1\,1} & c_{2\,1}x_{1\,2} & c_{2\,1}x_{1\,3} & c_{2\,1}x_{1\,4} & \cdots & c_{2\,1}x_{1\,42} \\ c_{3\,1}x_{1\,1} & c_{3\,1}x_{1\,2} & c_{3\,1}x_{1\,3} & c_{3\,1}x_{1\,4} & \cdots & c_{3\,1}x_{1\,42} \\ c_{4\,1}x_{1\,1} & c_{4\,1}x_{1\,2} & c_{4\,1}x_{1\,3} & c_{4\,1}x_{1\,4} & \cdots & c_{4\,1}x_{1\,42} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{42\,1}x_{1\,1} & c_{42\,1}x_{1\,2} & c_{42\,1}x_{1\,3} & c_{42\,1}x_{1\,4} & \cdots & c_{42\,1}x_{1\,42} \end{bmatrix} \tag{3.15}$$

The coefficients in equation (3.15) are extracted in matrix notation to form equation (3.16)

$$[DNN\ model\ weight\ matrix] = \begin{bmatrix} c_{1\,1} & c_{1\,1} & c_{1\,1} & c_{1\,1} & \cdots & c_{1\,1} \\ c_{2\,1} & c_{2\,1} & c_{2\,1} & c_{2\,1} & \cdots & c_{2\,1} \\ c_{3\,1} & c_{3\,1} & c_{3\,1} & c_{3\,1} & \cdots & c_{3\,1} \\ c_{4\,1} & c_{4\,1} & c_{4\,1} & c_{4\,1} & \cdots & c_{4\,1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{42\,1} & c_{42\,1} & c_{42\,1} & c_{42\,1} & \cdots & c_{42\,1} \end{bmatrix} \quad (3.17)$$

$$DNN\ model\ weight\ matrix = [iw_{ij}]'[lw_{ij}]' \quad (3.18)$$

Where;

iw_{ij} = input weight matrix,

lw_{ij} = layer weight matrix

3.5 Formulation of Algorithm

Figure3.3 present a flowchart for the proposed deep neural network learning algorithm for cloud botnet detection, the composition of the flowchart are the preprocessed dataset employed in this research, feed into the defined model and its configuration as expressed in the preceding subsection, next the hyper-parameter set range is established, defined epoch is run alongside the batch size, an optimizer is applied for optimality determination, performance optimality is determined for cloud botnet detection, else if no optimal performance, hyper-parameter bound is set again for further training and performance determination.

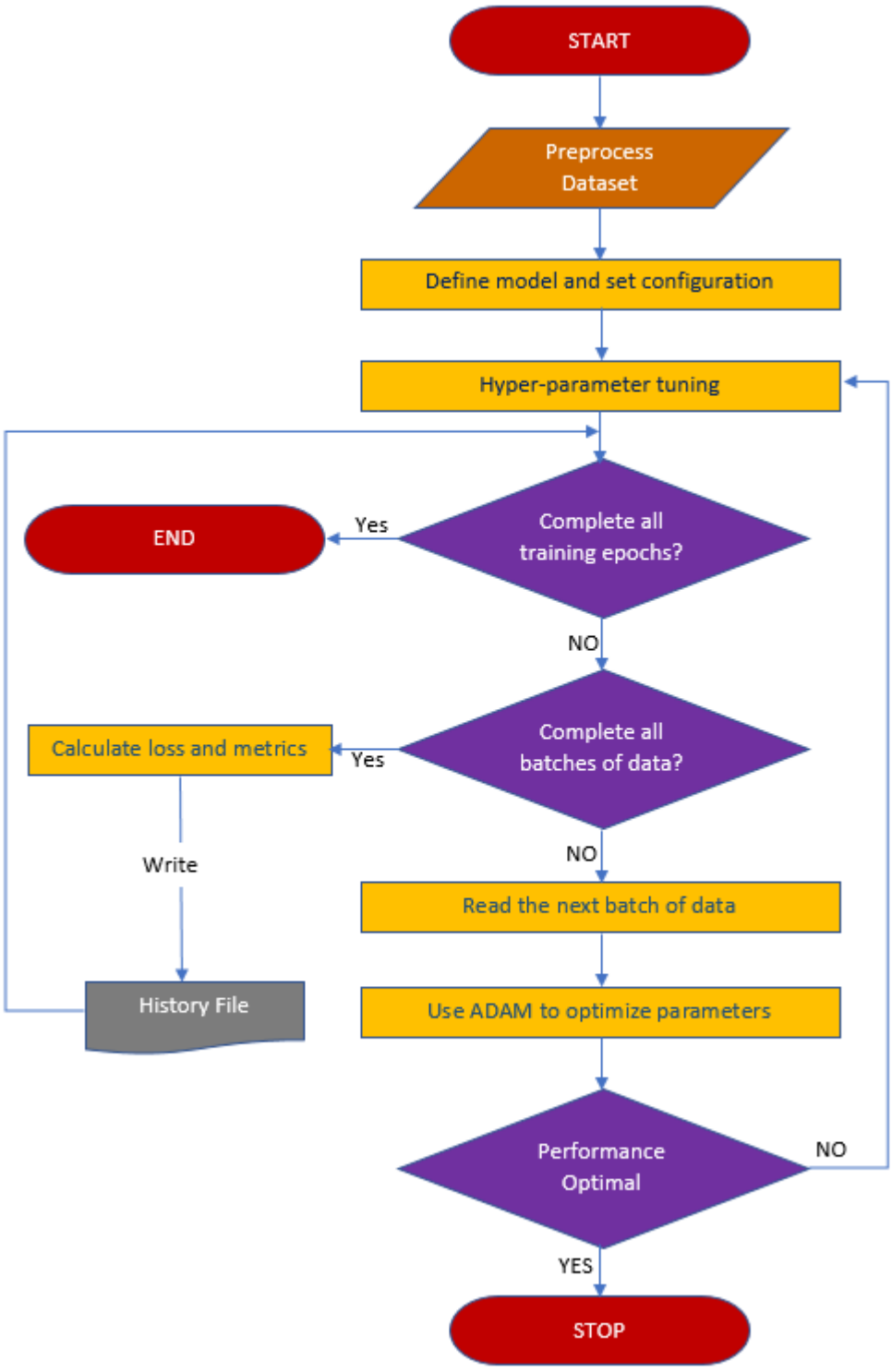

Figure 3.3 Flowchart forproposed deep neural network learning algorithm for cloud botnet detection

3.6Validation of the Proposed Model Through Experimentation

The Figure 3.4 presents the experimental procedure of the proposed DNN, at the initial phase adopted dataset obtained goes through the preprocessing phase which comprise of encoding and normalization for efficient machine training process and classification. The next phase is the test option which comprises of dataset splitting into the training, validation and testing options with 70% of the dataset for training, 15% for validation and 30% for testing. The preceding phase aids in hyper parameter tuning of the machine learning model as well as building efficient learning model, furthermore, the hyper parameters involve in this research for determination of optimal performance are the number of hidden layers, epoch, number of neurons and dropout. The next phase is the developed model with enhanced performance for cloud botnet classification using the testing dataset for evaluation of accuracy, sensitivity, False Alarm Rate (Far), And False Positive Rate (FPR).

The reported experimental in this research obtained its dataset from UCI KDD dataset repository, known as ISCX NSL-KDD dataset. The dataset has 42 attributes and 125973 instances, to adequately classify the NSL-KDD dataset.

3.6.1Datapreprocessing

Data preprocessing regards to an essential part of machine learning method, which presents dataset for better performance learning there by addressing inconsistencies impede the accurate and other related performance matrices I actualizing optimal learning process results. Inorder to make the dataset usable for the candidate model, this research employed data preprocessing techniques such as label encoding and data normalization, which are further outlined below;

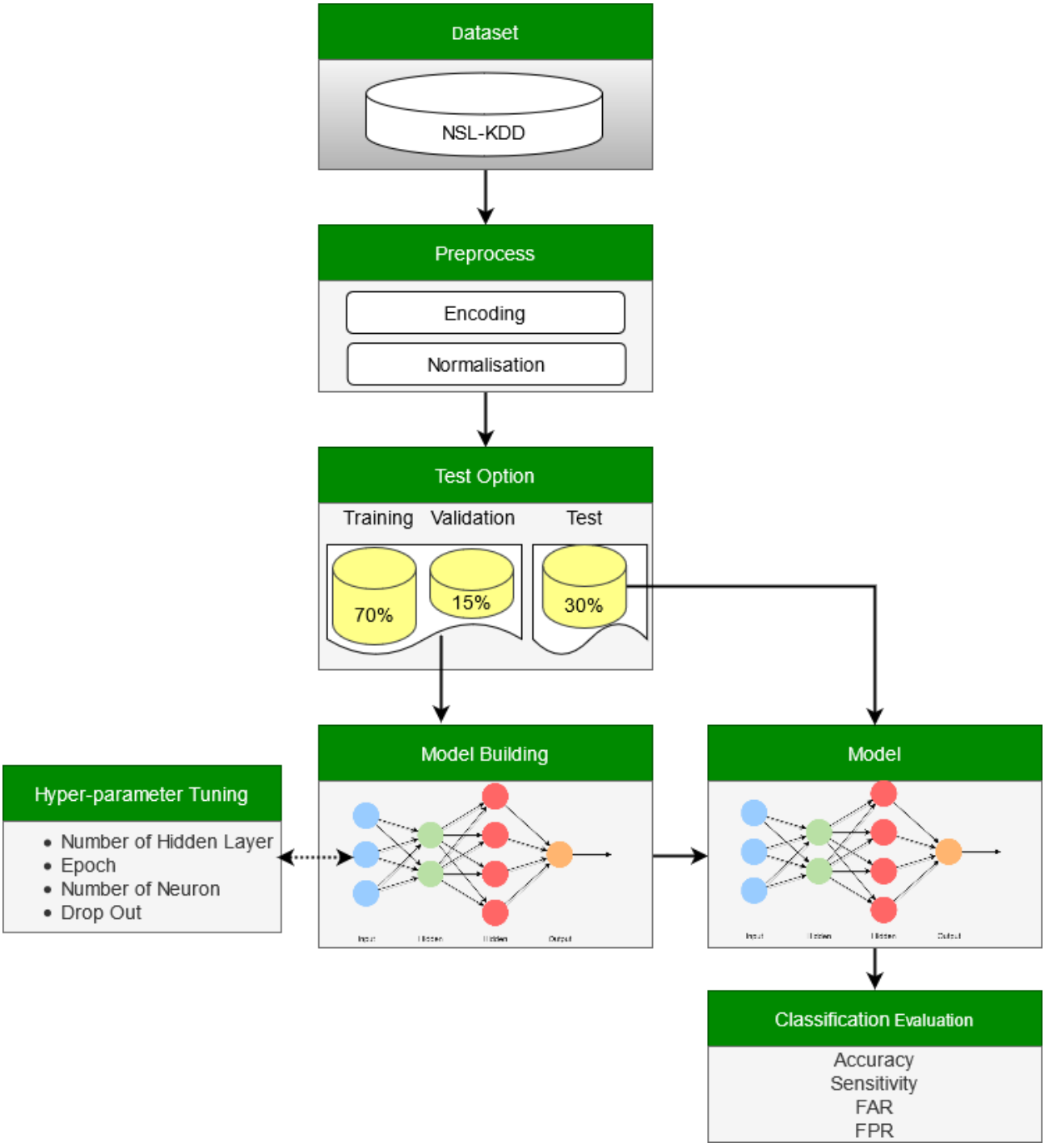

Figure 3.4 Architectural structure of the proposed deep neural network

3.6.2Label encoding

In other to enhance model performance and efficiency, this research encoded some of the features from one form toanother, such as class label which is either normal (benign) orabnormal (attack or intrusion) was encoded to 0 and 1 respectively, while theprotocolattribute was encoded to integer values such as tcp, udp, and icmp representing (1), (2) and (3) respectively.

3.6.3Data normalization

Researchers have it established that data normalization can generate an optimal performance in terms of machine learning algorithms such as DNN inclusively, normalization offers algorithms efficacy when the data is scaled in the following given range of [-1, 1] or [0, 1], based on this, the research employed the use of Min-Max normalization technique for data scaling, which implies that the research data is scaled between [0, 1]. Equation below presents the Min-Max normalization mathematical expression.

$$x_n = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.19}$$

Where:

x_n denotes the normalized sample new value

x denotes the value to be normalized

x_{min} denotes the minimum value

x_{max} denotes the maximum value

3.6.4 Modelbuilding

The proposed deep learning model for botnet attack detection in cloud is train using the training dataset in order to extract and learn the patterns that exist in the cloud botnet

dataset, the weight of the deep neural network is used to learn from the dataset that is feed into the network, at each layer complex pattern are learned from the computational processes in the hidden layer of the model, furthermore, the deep learning model consists of the input layer, hidden layer and output layer, each layer encompasses of neurons, while the neurons from a predecessor layer is connected to successor neurons in the next layer thereby building a connection known as weight network for transfer of the output between layers.

The proposed DNN learning algorithm for cloud botnet attack comprises of 5 layer architecture, the first layer is the input, the second, third and fourth layers are the hidden layer, while the fifth layer represent the output layer, the input layer compose of 42 neurons which equate to the numbers of the feature found in the adopted dataset, the hidden layers 2,3,4 which lies between the input and output layers are made up of 29 neurons each respectively, the processing of the input data for model building takes place in the hidden layer through weights manipulation, transforming the input data based on the a non-linear functions known as activation functions, the activation function are the mathematical technique that aids in controlling the output of a neural network(NN), the activation function adopted in this research are Rectified Linear Unit (ReLU) for the hidden layers and Sigmoid function for the output layer, equations (3.20) and (3.21) represent the mathematical equations respectively.

$$f(x) = \max(0, x)\,\{x_i, if\ x_i > 0, 0, if\ x_i < 0 \tag{3.20}$$

x: input dataset instances

i: total number of dataset instances

$$f(x) = \frac{1}{1+e^{(-x)}} \quad (3.21)$$

x: input dataset instance

While the fifth layer represent the output layer that generates the resultant of the prediction of the proposed model.

3.6.5 Hyperparameter tuning

The hyper parameters such as the number of layer, epoch, number of neuron, and dropout ratio are the parameters of a model that can be tune to achieve enhanced model performance for detection of botnet attack in cloud, the number of layers determines how deep the neural network is and effectiveness of the neural network, the epoch determines the iterations duration during the training process of the deep learning model for botnet attack detection, number of neurons per layer though rule of thumb does not exist for this as well as the number of layers both are determined based on permutation, while this research adopted from existing literatures based on performance established in literature, dropout ratio entails the technique that drops out the connections between neural network at each layer based on defined percentage ratio, the dropout takes place at random, that is weight connection that is dropped at each layer.

3.6.6 Model

The model in this aspect represents the final model obtained after training, the testing dataset is presented to the model in order to determine how well the model can generalize, while the performance is been evaluated with the following relevant evaluation measure; accuracy, sensitivity, FAR and FPR, which is also compared against exist related literature

to establish performance enhancement of the proposed deep learning model for botnet attack detection in cloud.

3.7Performance Evaluation Analysis

The research proposed to employ the performance evaluation based on the following metric;

1. **Confusion Matrix**:is a key parameter used in evaluating the performance of machine learning classifiers. The four major component of confusion matrix are true positive, false positive, true negative and false negative which is detailed in Figures 3.5 and 3.6respectively. In general, classification performance should be evaluated based on f-measure, precision, and recall.

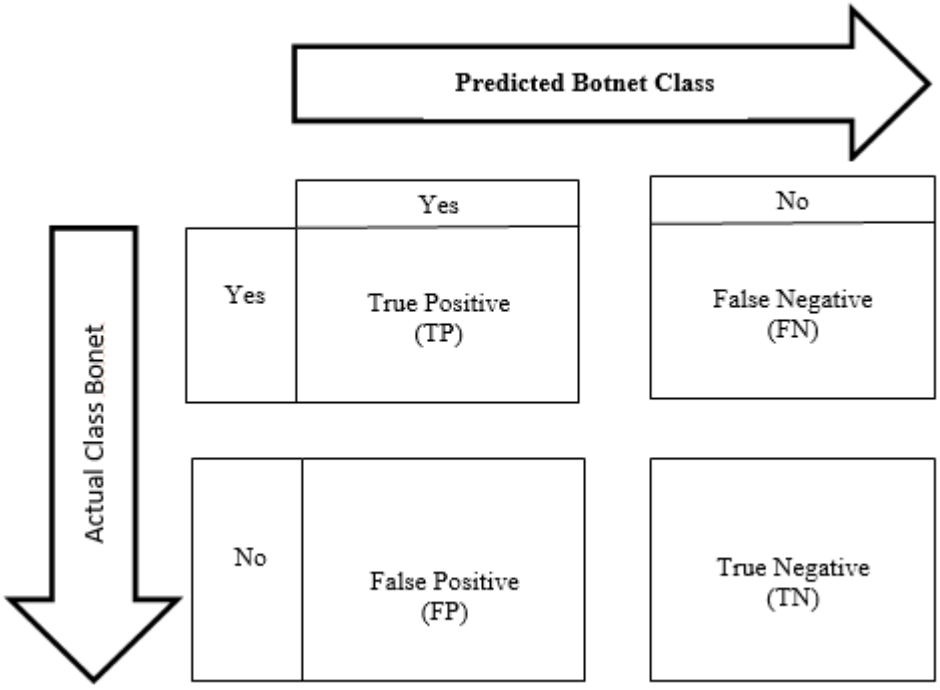

Figure.3.5 Confusion matrix measurement

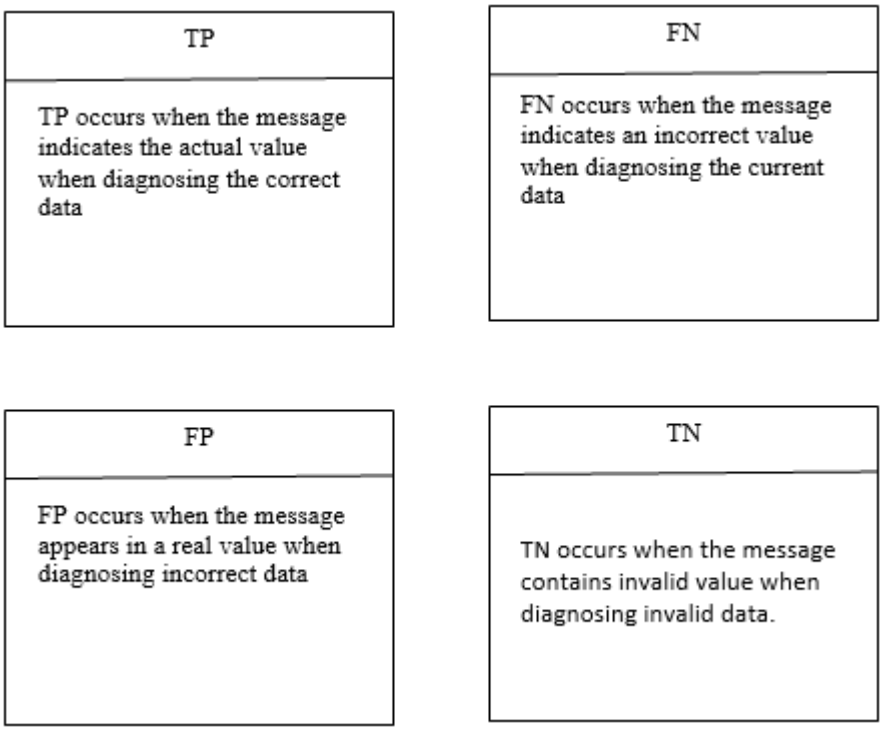

Figure 3.6 Definition of confusion matrix measurement

2. Accuracy:proportion of total number of correct predictions

$$\frac{TP+TN}{P+N} \tag{3.22}$$

3. Recall (Sensitivity):proportion of positives correctly predicted as positive

$$\frac{TP}{P} \tag{3.23}$$

4. False Alarm Rate (FAR):this defines the numbers normal cloud botnet instances classified as an attack by the total number of normal instances in the testing dataset.

$$FAR = \frac{FP}{FP+TN} \tag{3.24}$$

5. False Positive Rate (FPR): this defines the percentage of normal instances incorrectly classified

$$FPR = \frac{FP}{TN+FP} \tag{3.25}$$

CHAPTER FOUR

4.0 RESULTS AND DISCUSSION

This section analyses the result output generated from the proposed DNN learning algorithm for cloud botnet attack detection in this research.

4.1 Results for DNN Learning Algorithm for Cloud Botnet Attack Determination

In other to establish the parameters used in this research, such as numbers of hidden layers as well as the epoch value employed, the numbers of hidden layers was varied within a defined range of 1 to 3 hidden layer Su *et al*. (2020), same goes to the epoch value of 300Ahmed*et al.*(2020), which is based on performance excellent recorded in related literatures, the loss function and optimizer which are binarycross entropy and ADAM was based on the nature of experiment been performed in this research which is a cloud botnet detection,meanwhile, the next section presents the experiment performed in other to identify the epoch value that generates the result with enhance performance for cloud botnet attack detection based DNN as well as establishing the perform function of the hidden layer after variation as presented inTable 4.1 outlining the hidden layer variation in order to determine the number of hidden layer that will generate optimal performance within the range of 1 to 3 hidden layer for the proposed DNN learning algorithm for cloud botnet attack detection.

4.1.1Generated training curves

The training curve help obtained in this research process analysis to determine the optimality performance of the internal parameters over an incremental iterative process of

model training (epoch), for a better evaluation performance to take place, indicating that the training process of the model is learning perfectly, the curve will tends to 0.00 loss value, with minimal or no overfitting, however, due to the noticed overfitting, a dropout value of 0.1, 0.2 and no dropout value was applied on the proposed model, while the epoch value across was 300,Figures 4.1, 4.2 and 4.3 present the model loss curve graph.

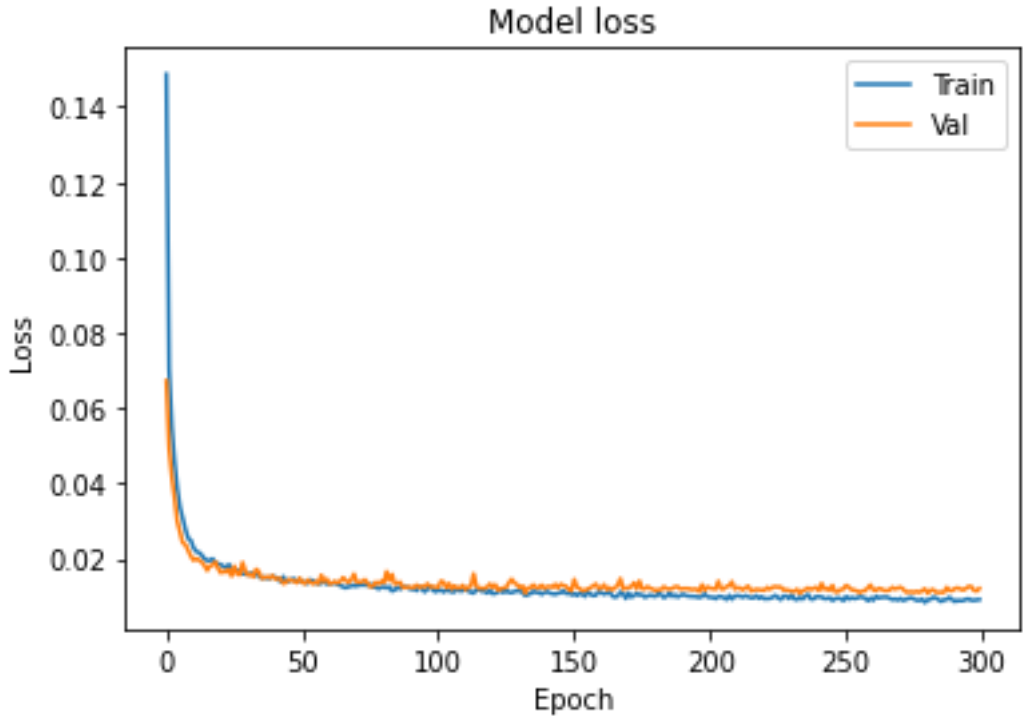

Figure 4.1 Model Loss Curve with Dropout 0.1

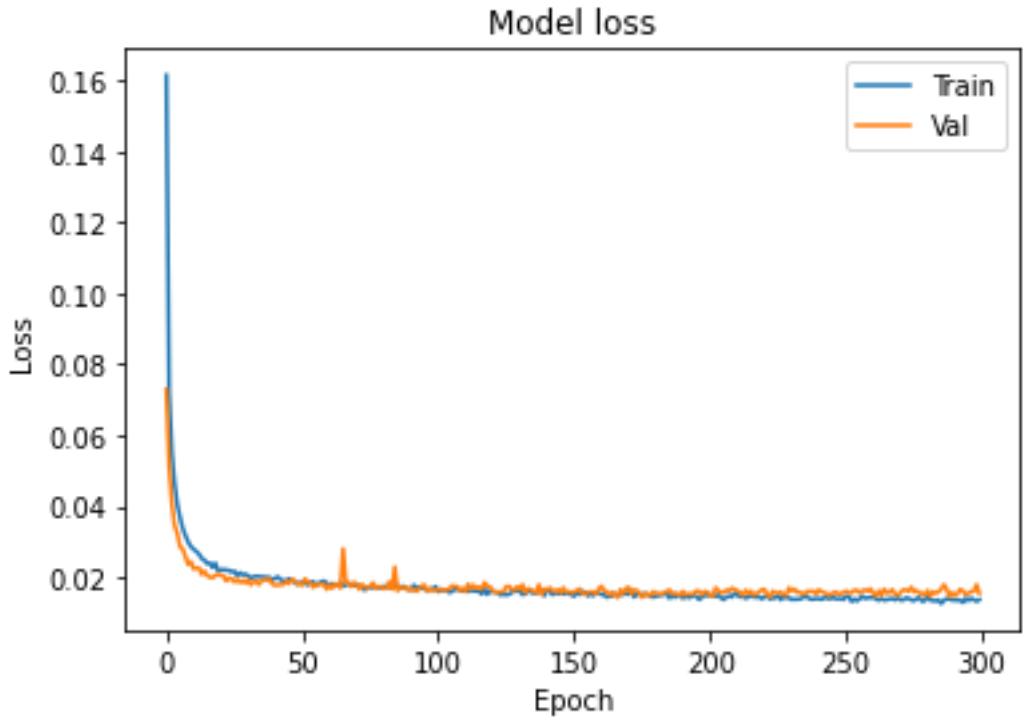

Figure 4.2 Model Loss Curve with Dropout 0.2

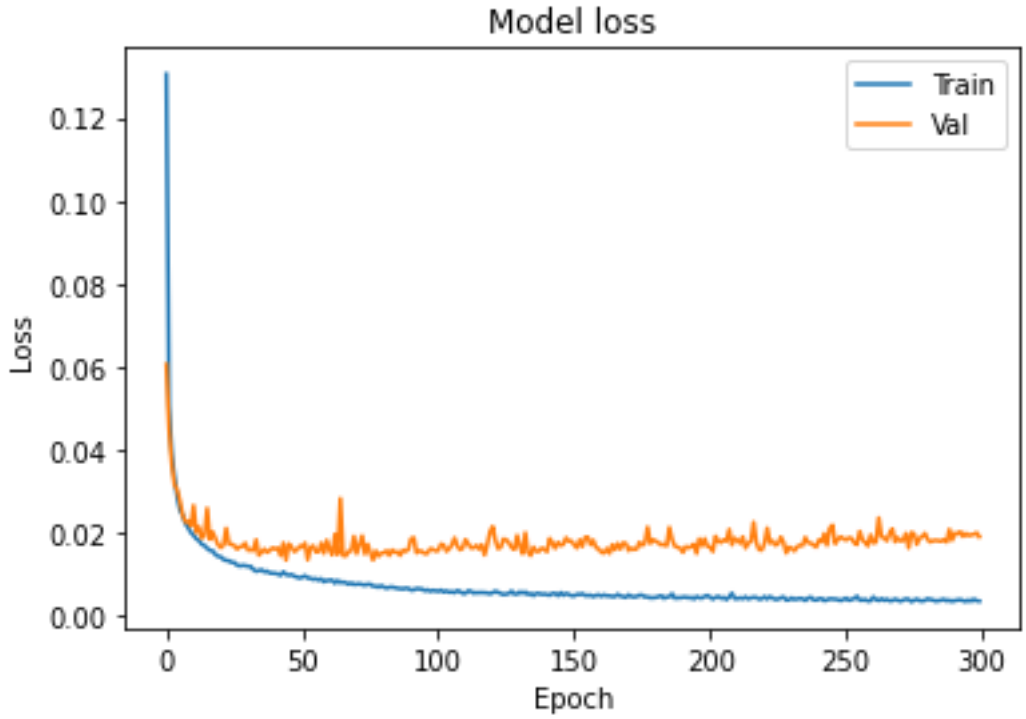

Figure 4.3 Model Loss Curve with No Dropout

4.1.2 Model loss curve analysis

The presented model loss curves in Figures 4.1, 4.2, 4.3 indicates how well theperformance model learns, with variation of parameter tuning as well as default performance output showing a clear indication of over fitting, as noticed, the validation and training plot are not correlated, with the training loss declining while the validation loss remains constant over the epoch, reflecting that the model is not improving but rather overfitting the training data.

Meanwhile, Figures 4.1 and 4.2 respectively indicates a clearly health correlation between the validation and training loss, which seems to be decreasing at a steady pace with a constant value, reflecting that the model is well trained as well as equally good on the training data and also the unseen data (validation dataset), however, the Figure 4.1 is regarded to as the best output due its smooth curve.

4.1.3 Accuracy training curve

The model accuracy curve indicates performance excellent when the plot increases towards the score of 100 over an epoch value, representing a well-trained model that learns with better performance, devoid of over fitting, however, and over fitting was experienced leading to parameter turning by dropout function, Figures 4.4, 4.5 and 4.6 present the plots for accuracy training curves with dropout values 0.1, 0.2 and no dropout respectively.

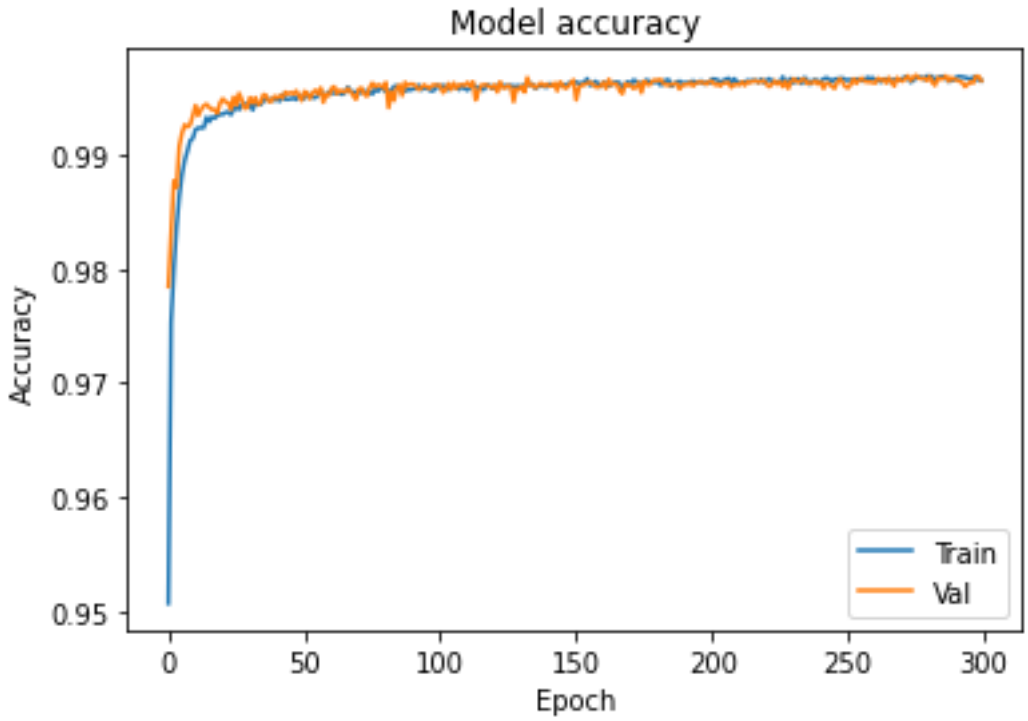

Figure 4.4 Training Accuracy Curve with Dropout 0.1

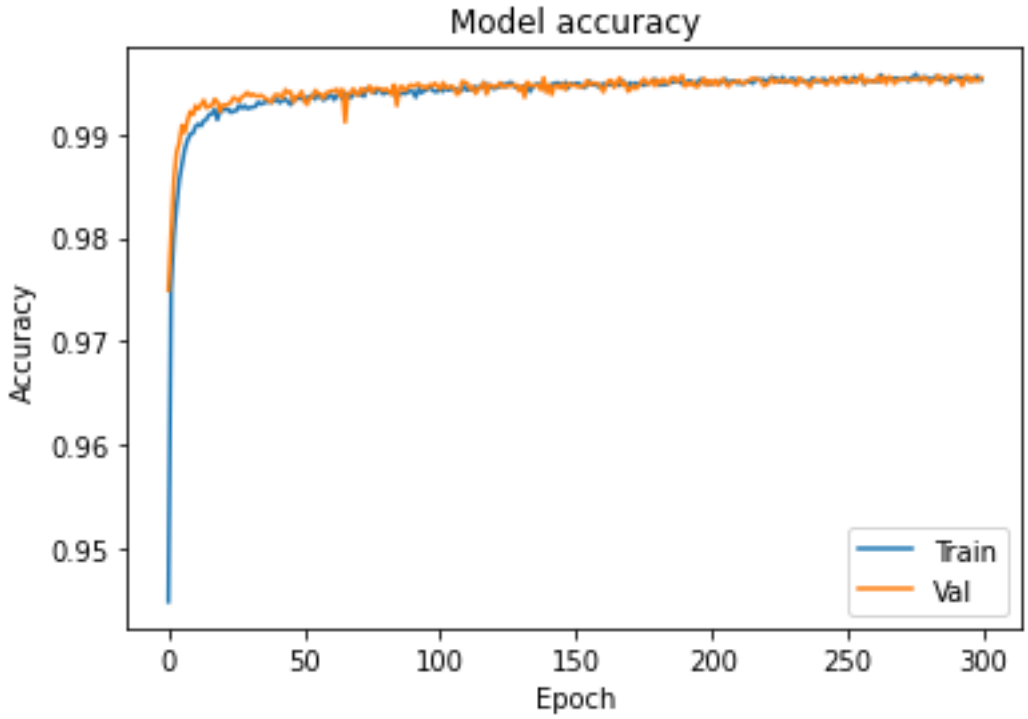

Figure 4.5 Training Accuracy Curve with Dropout 0.2

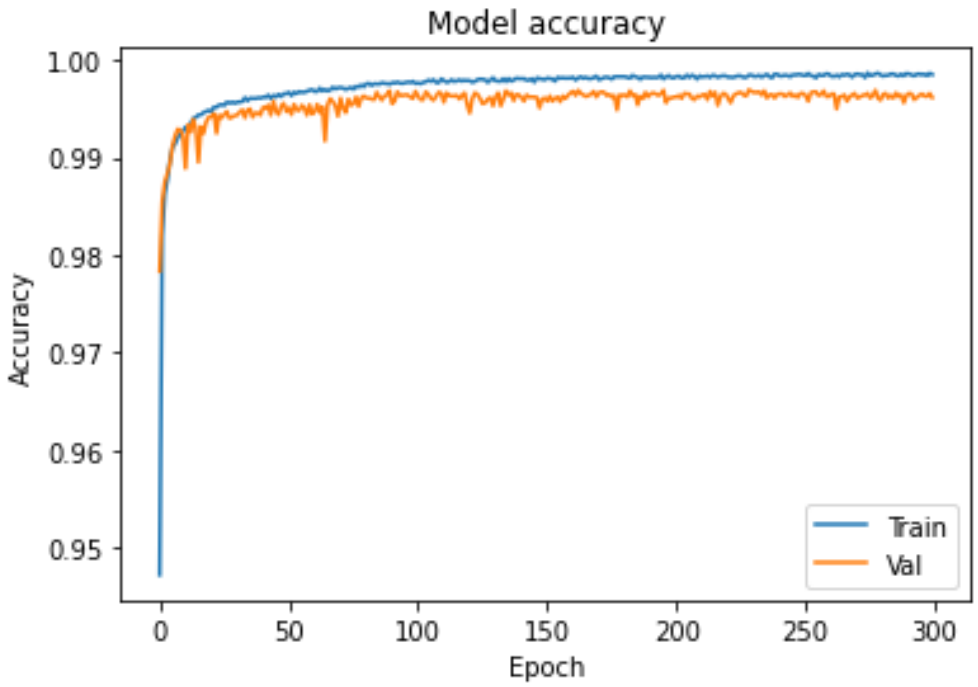

Figure 4.6 Training Accuracy Curve with No Dropout

4.1.4 Training accuracy curve analysis

An indication from Figure 4.6 no dropout depicts a clear overfitting as the model training and validation loss do not correlate, while Figures 4.4 and 4.5 reflect a health correlation amongst the training and validation loss, both increases steadily which also connotes model performance rating as good on both the training and validation dataset (unseen dataset), however, Figure 4.4 is more preferably based on its smoothness compared to Figure 4.5, while throughout the training experiment the epoch value of 25 was identified as most preferred epoch based on the model learning performance in term of a well generalization performance as well as accuracy performance, furthermore, Table 4.1 identify the optimal number of hidden layer for the proposed DNN for cloud botnet attack detection.

Table 4.1**Performance Measure of Hidden Layer Variation**

Number of hidden layer(s)	**1**	**2**	**3**
Accuracy	99.40	99.53	99.62

From Table 4.1, a steady increase in accuracy performance is experienced as the number of hidden layers graduates from a hidden layer, two hidden layer and then to three hidden layer, an exponential increase in accuracy is attained in the DNN with the architecture of DNN having three hidden layer recording accuracy of 99.62 which is an optimal performance compared to DNN architecture of hidden layers one and two which recorded an accuracy of 99.40 and 99.53 respectively, it can be noticed from Table 4.1, that as the number of hidden layer increases so also the accuracy performance increases, which depicts that the more the numbers of hidden layers the more the DNN is able to learn hidden complex pattern in the dataset with an enhanced performance, based on this outcome this research adopted three hidden layer parameter for the proposed DNN learning algorithm for cloud botnet attack detection.

The Table 4.2 present the parameters deployed for the proposed DNN learning algorithm for cloud botnet attack detection.

Table 4.2**Deployed Parameters**

Optimizer	**Loss Function**	**Metrics**	**Batch_size**	**Epoch**
Adam	Binary-Cross entropy	Accuracy	128	25

4.2 Result analysis

This section discus the performance achieved from the proposed DDN learning algorithm for cloud botnet attack detection, based on the following performance metric; accuracy, sensitivity, FAR, FPR, and detection rate, also

4.2.1 Results

The Table 4.3 present the scores achieved based on the performance metrics; accuracy, sensitivity FAR, FPR and detection rate by the proposed model for cloud botnet attack detection.

Table 4.3**Performance Score of the Proposed Model**

	Accuracy	**Sensitivity/TPR/DR**	**FAR**	**FPR**
DNN Learning Algorithm (Proposed Model)	0.9965	0.9968	0.0035	0.004

The achievement recorded from the proposed DNN learning algorithm model indicates that the proposed model was capable of consistent cloud botnet attack detection with high confidence as experienced from the accuracy and sensitivity score rate, as each score well above 0.9900, while a low FAR and FPR was achieved, with FAR have as low as 0.003 score rate.

An accuracy of 0.9965 was achieved which reflects the high strength of the proposed model in distinguishing a cloud botnet attack from a normal internet traffic flows, in same manner sensitivity score of 0.9968 was achieved from the proposed model, while 0.0035 and 0.004 was achieved for FAR and FPR respectively.

4.3 Comparison of Proposed Model and Benchmark Literature

In order to establish the optimal capability of the proposed DNN learning algorithm model against the existing literature model performance, the same dataset used by the benchmark journals was adopted in this research work, for the purpose of having a fair level of comparison.

4.3.1 Accuracy comparison

The evidence from the outcome of the performance score in terms of accuracy obtained from the analysis performed with the proposed DNN learning algorithm indicates as presented in Figure 4.7 that the proposed model in this research have a better accuracy score of 0.9965 outperforming the accuracy achieved by Tuan*et al*, 2019 and Azzaoui *et al*. (2021), which recorded an accuracy of 0.98808 and 0.9963 respectively.

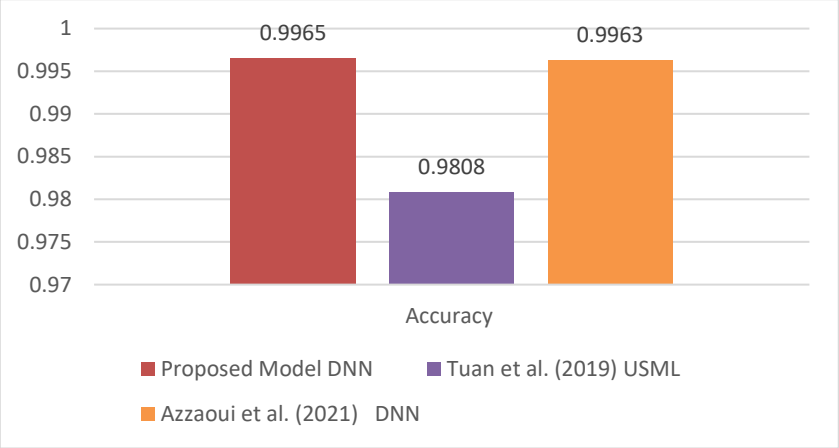

Figure 4.7 Accuracy Based Comparison

4.3.2 Sensitivity comparison

The sensitivity score which is also known as recall, detection rate as well as TPR achieved 0.9968 from the proposed model hence outperforming the sensitivity scores of the research

carried out by Tuan*et al*, 2019 and Azzaoui*et al.* (2021), with recorded scores of 0.9188 and 0.8650 respectively presented in Figure 4.8, this is a clear optimal performance indication of the proposed model capability in cloud botnet attack detection accurately.

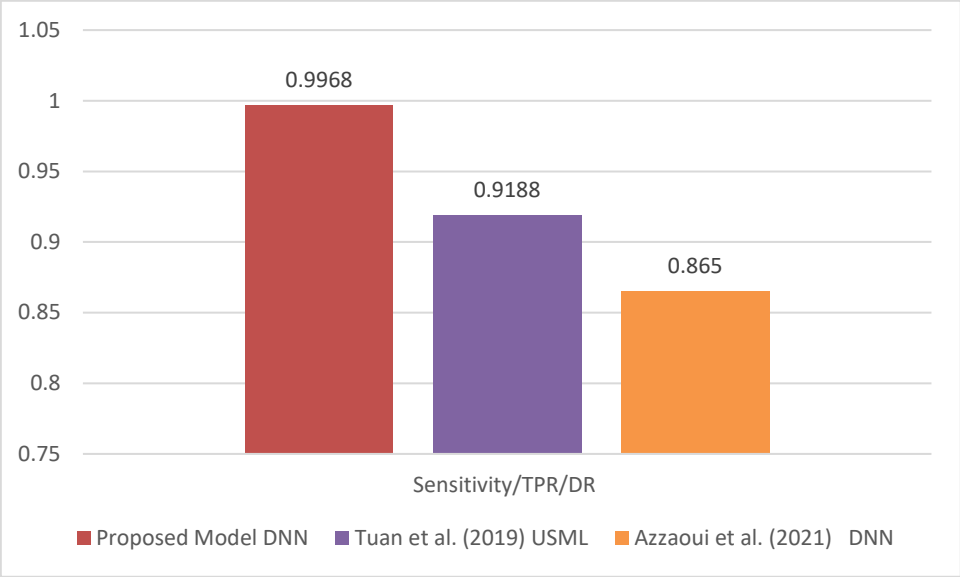

Figure 4.8 Sensitivity Based Comparison

4.3.3 False alarm rate (FAR)

The proposed model FAR score achieved 0.0035 which indicate the lowest as compared to the benchmark journal of Tuan*et al.*, 2019 which score 0.0192 with a variation of 0.0157, the observation from the achieved FAR score also represents the performance strength of the proposed model, Figure 4.8 present the FAR based comparison.

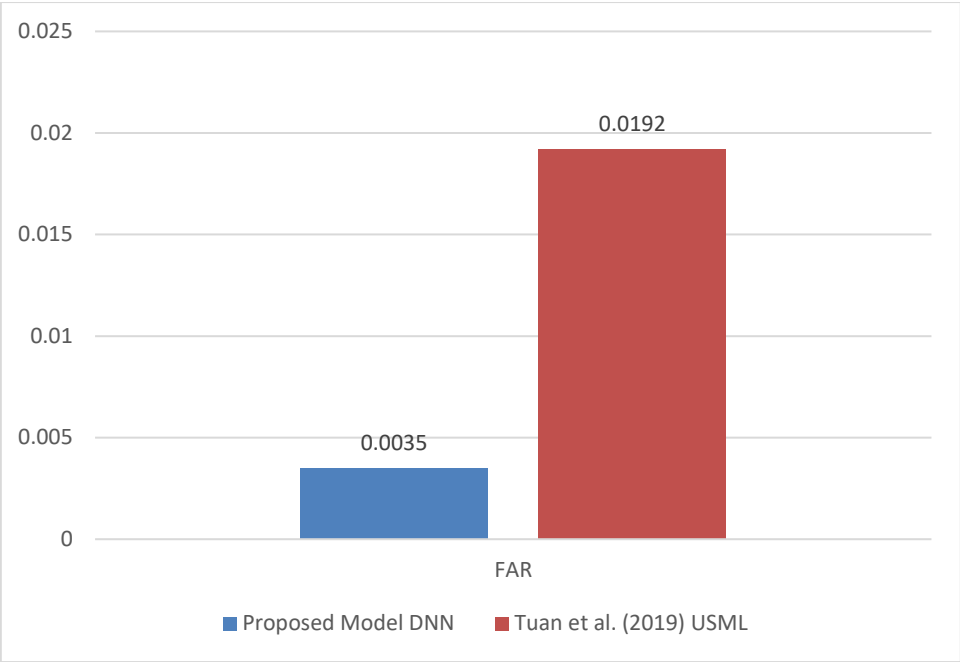

Figure 4.9 FAR Based Comparison

4.3.4 False positive rate

The FPR of the proposed model against the benchmark journal achievement indicates the lowest FPR of 0.004 against 0.9188 and 0.11 for the research work by Tuan*et al*. 2019 and Azzaoui *et al*. (2021), respectively, while the FPR score of Tuan*et al*. 2019, is the highest follow by Azzaoui*et al*. (2021), research work FPR score, Figure 4.9 present the FPR based comparison.

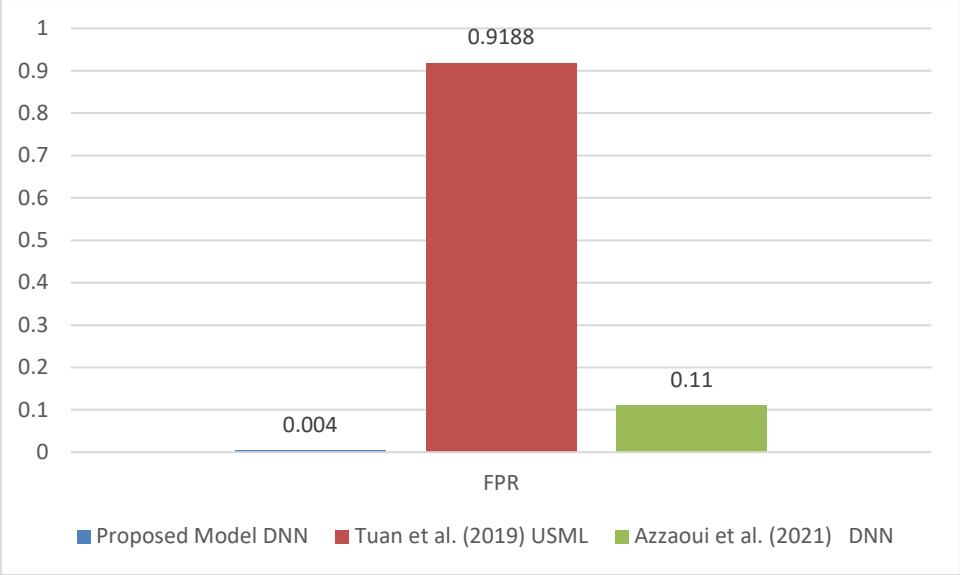

Figure 4.10FPR Based Comparison

Furthermore, Table 4.4 present a comparison of the proposed model with the benchmark journal in terms of the techniques employed by the benchmark journals, a kin observation at Table 4.4, shows that the proposed DNN learning algorithm model outperform other techniques used by existing literature such as the benchmark journal for this research, as indicated in the performance metrics scores of accuracies, sensitivity, FAR and FPR.

Table 4.4**Technique performance metric comparison**

Reference	Method	Accuracy	Sensitivity/TPR/DR	FAR	FPR
Proposed Model	**DNN**	**0.9965**	**0.9968**	**0.0035**	**0.004**
Tuan *etal*. (2019)	USML	0.9808	0.9188	0.0192	0.9188
Azzaoui *et al*. (2021)	DNN	0.9963	0.865	-	0.11

Indications from Table 4.4 reveals that one of the most common performance measures in the field of botnet detection is the accuracy, sensitivity as well as FPR as they appear in all the benchmark literature performance metric, the technique employed by benchmark literature are Unsupervised Machine Learning (USML) and DNN by Tuan*et al*. 2019 and Azzaoui *et al.*(2021), respectively despite the good performance score achieved by these benchmark literatures, a high FAR and FPR is one of the setback as can be seen from Table 4.4, however, impeding on optimal performance of detection capability, therefore, the proposed DNN learning algorithm for cloud botnet attack detection have proven its supremacy over existing literatures by outperforming other techniques employed in terms of the relevant performance metric used based on proper dataset scaling and encoding as well as hyper parameter tuning for better performance.

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATION

5.1 Conclusion

This research work was motivated by the swelling rate of botnet attacks using DDoS on cloud environments trans versing the earth sphere and the understanding from the literature review of the availability of classification algorithms that have not been compared in terms of their performance on botnet datasets. In this paper, a deep neural network model is suggested which consists of five layers and is tuned using hyper- parameters to choose the best set of results. NSL-KDD datasets were used in this experiment. The research conducted experiments on the same dataset that has been used by most of the previous works, to give this research work an enhance accuracy performance. Furthermore, the performance of the proposed DNN model achieved an average accuracy of 0.9965, sensitivityscore(Recall) of 0.9968, while 0.035 and 0.004 was also achieved for false alarm rate(FAR), and false positive rate(FPR) respectively.

5.2 RECOMMENDATION

For the future work, this research recommends anexpected exploration ofbroaden range of hyper-parameters, likewise explore other deeper neural network field of study in order to improve model performance evaluation.

5.3 CONTRIBUTION TO KNOWLEDGE

This research achieved the following which serves as acontribution to knowledge;

1. An architecture for an enhanced classification of cloud botnet attack

2. An optimized model for the classification of cloud botnet attack with a performance scores of 0.9965, 0.9968, 0.0035 and 0.004 for accuracy, sensitivity, FAR and FPR respectively

REFERENCES

Abdulhamid, S. I. M., Shuaib, M., Osho, O., Ismaila, I., & Alhassan, J. K. (2018). Comparative analysis of classification algorithms for email spam detection. *International Journal of Computer Network & Information Security*, 10(1), 60–67. doi:10.5815/ijcnis.2018.01.07

Ahmed, A. A., Jabbar, W. A., Sadiq, A. S., & Patel, H. (2020). Deep learning-based classification model for botnet attack detection. *Journal of Ambient Intelligence and Humanized Computing*, 1-10. doi: 10.1007/s12652-020-01848-9

Ali, A., Waheb, A., Ali, A. J., Sadiq, S., & Patel, H. (2020). Deep learning - based classification model for botnet attack detection based attack that seeks to subvert multiple computers. *Journal of Ambient Intelligence and Humanized Computing*, 1-10, doi: 10.1007/s12652-020-01848-9

Alieyan, K., Almomani, A., Anbar, M., Alauthman, M., Abdullah, R., & Gupta, B. B. (2019). Domian name serverrule-based schema to botnet detection. *Enterprise Information Systems*, 15(4), 545-564. doi: 10.1080/17517575.2019.1644673

Almalaq, A., & Edwards, G. (2017). A review of deep learning methods applied on load forecasting. *In16th International Conference on Machine Learning and Applications (ICMLA)*, (pp. 511-516).Cancun, Mexico: IEEE.

Alomari, E., Manickam, S., B. Gupta, B., Karuppayah, S., & Alfaris, R. (2012). Botnet-based distributed denial of service attacks on web servers: Classification and art. *International Journal of Computer Applications*, 49(7), 24–32. doi: 10.5120/7640-0724

Amini, P., Azmi, R., & Araghizadeh, M. (2014). Feature selection on persian fonts: A comparative analysis on genetic annealing algorithm, guided evolution simulated annealingand genetic algorithm.*Procedia Computer Science*, 3, 1249-1255. doi: 10.1016/j.procs.2010.12.200

Antonakakis, M., April, T., Bailey, M., Bursztein, E., Cochran, J., Durumeric, Z., Alex Halderman, J., Menscher, D., Seaman, C., Sullivan, N., Thomas, K., Zhou, Y., (2017). Understanding thr mirai botnet. *In the Proceedings of the 26thSecurity Symposium*, 17, (1093-110). doi: 10.5555/3241189.3241275

Asad, M., Asim, M., Javed, T., Beg, M. O., Mujtaba, H., & Abbas, S. (2019).Deepdetect: Detection of distributed denial of service attacks using deep learning. *The Computer Journal*, 63(7), 71780-71790.doi: 10.1093/comjnl/bxz064

Awad, A. A., & Sayed, S. G. (2019). Collaborative framework for early detection of remote access trojans-bots attacks. *Institute Electrical and Electronic Engineer Access*, 7, 71780–71790. doi: 10.1109/ACCESS.2019.2919680

Azzaoui H., Boukhaml, A. Z. E., Arroyo Guardeño, D., & Bensayah, A. (2021).

Developing new deep-learning model to enhance network intrusion classification. *Evolving Systems*, 1-9. doi: 10.1007/s12530-020-09364-z

Bandara, K. R. W. V, Abeysinghe, T. S., Hijaz, A. J. M., Darshana, D. G. T., Aneez, H., Kaluarachchi, S. J., Sulochana, K. V. D. L., & Dhishandhammearatchi, M. (2016). Preventing distributed denial of service attack using data mining algorithms. *International Journal of Scientific and Research Publications*, 6(10), 390. retrieved from http://www.ijsrp.org/research-paper-1016/ijsrp-p5857.pdf

Bengio, Y. (2009). *Learning deep architectures for AI*. Boston, United States: Amazon Incorperated.

Booz, J., McGiff, J., Hatcher, W. G., Yu, W., Nguyen, J., & Lu, C. (2018). Tuning deep learning performance for android malware detection. *In 19th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, (pp. 140-145).Busan, Korea: IEEE.

Booz, J., McGiff, J., Hatcher, W. G., Yu, W., Nguyen, J., & Lu, C. (2019). Towards deep learning-based approach for detecting Android malware. *International Journal of Software Innovation (IJSI)*, 7(4), 1-24. doi: 10.1109/SNPD.2018.8441127

Carlin, A., Hammoudeh, M., & Aldabbas, O. (2015). Defence for distributed denial of service attacks in cloud computing. *Procedia Computer Science*, 73, 490–497. doi: 10.1016/j.procs.2015.12.037

Chen, J., Zeng, G. Q., Zhou, W., Du, W., & Lu, K. D. (2018). Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization. *Energy Conversion and Management*, 165, 681-695. doi:10.1016/j.enconman.2018.03.098

Cogranne, R., Doyen, G., Ghadban, N., & Hammi, B. (2017). Detecting botclouds at large scale: A decentralized and robust detection method for multi-tenant virtualized environments.*Transactions on Network and Service Management*, 15(1), 68-82.doi: 10.1109/TNSM.2017.2785628

Deka, R. K., Bhattacharyya, D. K., & Kalita, J. K. (2017). Ddos attacks: Tools, mitigation approaches, and probable impact on private cloud environment. Big Data Analytics for Internet of Things, 285-319.doi: 1710.08628v1

Fadhlee, R., Dollah, M., Faizal, M. A., Arif, F., Zaki, M., & Xin, L. K. (2018).Machine learning for hyper text transfer protocolbotnet detection using classifier algorithms. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10(1-7), 27-30. 10(1),27–30. Retrieved from https://web.archive.org/web/20180413211821id_/http://journal.utem.edu.my/index.php/jtec/article/viewFile/3591/2484

Feily, M. (2009). A survey of botnet and botnet detection. *In Third International*

Conference on Emerging Security Information, Systems and Technologies, (pp. 268-273). Athens, Greece: IEEE.

François, J., Wang, S., Bronzi, W., State, R., Engel, T., François, J., Wang, S., Bronzi, W., State, R., Engel, T., & Detecting, B. (2012). Botcloud: Detecting botnets using mapreduce. *InInternational Workshop on Information Forensics and Security*, (pp. 1-6). Iguacu Falls, Brazil: IEEE.

Hoang, X. D. (2018). Botnet detection based on machine learning techniques using domain name system query data. *Future Internet*, 10(5), 43. doi: 10.3390/fi10050043

Homayoun, S., Ahmadzadeh, M., Hashemi, S., Dehghantanha, A., & Khayami, R. (2018). BoTShark: A deep learning approach for botnet traffic detection. *In Cyber Threat Intelligence*, 70, 137–153. doi: 10.1007/978-3-319-73951-9_7

Idhammad, M., Afdel, K., & Belouch, M. (2018). Detection system of hyper text transfer protocol distributed denial of serviceattacks in a cloud environment based on information theoretic entropy and random forest. *Security and Communication Networks*. doi:10.1155/2018/1263123

Khundrakpam, S. &Tanmay, D. (2020). Efficient classification of Distributed denial of service attacks using an ensemble feature selectionalgorithm. *Journal of Intelligent Systems*, 29(1), 71-83. doi: https://doi.org/10.1515/jisys-2017-0472

Kudugunta, S., & Ferrara, E. (2018). Deep neural networks for botnet of detection. *Information Sciences*, 467, 312-322. doi: https://doi.org/10.1016/j.ins.2018.08.019

Lin, H., Sun, Q., & Chen, S. Q. (2020). Reducing exchange rate risks in international trade: a hybrid forecasting approach of complete eensemble eempirical mode decomposition and multilayer long short term memory. *Sustainability*, *12*(6), 2451. doi: https://doi.org/10.3390/su12062451

Lin, K. C., Chen, S. Y., & Hung, J. C. (2014). Botnet detection using support vector machines with artificial fish swarm algorithm. *Journal of Applied Mathematics*, doi: 10.1155/2014/986428

Lu, W., Miller, M., & Xue, L. (2017). Detecting command and control channel of botnets in cloud. *In International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, 1, 55–62.Vancouver, Canada: Springer.

Mahardhika, Y. M., Sudarsono, A., & Barakbah, A. R. (2017, September). An implementation of Botnet dataset to predict accuracy based on network flow model. *In International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC)*, (pp. 33-39).Surabaya, Indonesia: IEEE

Mazini, M., Shirazi, B., & Mahdavi, I. (2019). Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and adaboost algorithms. *Journal*

of King Saud University-Computer and Information Sciences, 31(4), 541-553. doi: 10.1016/j.jksuci.2018.03.011

Nezhad, S. M. T., Nazari, M., & Gharavol, E. A. (2016). A novel DoS and DDoS attacks detection algorithm using ARIMA time series model and chaotic system in computer networks. *Institute Electrical and Electronics Engineers Communications Letters, 20*(4), 700-703. doi:10.1109/LCOMM.2016.2517622

Niazazari, I., Ghasemkhani, A., Liu, Y., Biswas, S., Livani, H., Yang, L., & Centeno, V. (2020). Deep neural network based wide-area event classification in power systems. *Arxiv Preprint arXi,* 2008.10151. Retrieved from https://arxiv.org/ftp/arxiv/papers/2008/2008.10151.pdf

Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 25). San Francisco, United States: Determination Press

Nur, W. A. N., Ibrahim, H., & Anuar, S. (2021).Multilayer framework for botnet detection using machine learning algorithms. *Institute of Electrical and Electronics Engineers Access,* 9, 48753-48768. doi:10.1109/ACCESS.2021.3060778

Pekta, A. (2018). Botnet detection based on network flow summary and deep learning. *International Journal of Network Management, 28(6), e2039.*doi: https:10.1002/nem.2039

Priyadarshini, R., & Barik, R. K. (2019). A deep learning based intelligent framework to mitigate distributed denial of service attack in fog environment. *Journal of King Saud University Computer and Information Sciences*. doi: 10.1016/j.jksuci.2019.04.010

Shenfield, A., Day, D., & Ayesh, A. (2018). Intelligent intrusion detection systems using artificial neural networks. *Information Communication Technology Express, 4*(2), 95-99. doi 10.1016/j.icte.2018.04.003

Silva, S. S., Silva, R. M., Pinto, R. C., & Salles, R. M. (2013). Botnets: A survey. *Computer Networks*, 57(2), 378-403. doi: 10.1016/j.comnet.2012.07.021

Singh, K. J., & De, T. (2017). Efficient classification of distributed denial of service attacks using an ensemble feature selection algorithm. *Journal of Intelligent Systems,* 29(1), 71-83. doi: 10.1515/jisys-2017-0472

Somani, G., Gaur, M. S., Sanghi, D., Conti, M., & Buyya, R. (2017). Distributed denial of service attacks in cloud computing: Issues, taxonomy, and future directions. *Computer Communications, 107, 30-48.* doi:10.1016/j.comcom.2017.03.010

Su, X., Shi, W., Qu, X., Zheng, Y., & Liu, X. (2020). DroidDeep: using Deep Belief Network to characterize and detect android malware. *Soft Computing*, *24*(8), 6017-6030. doi: 10.1007/s00500-019-04589-w

Syahirah Abdullah, R., Faizal Abdollah, M., Azri Muhamad Noh, Z., Zaki Mas, M.,

Rahayu Selamat, S., & Yusof, R. (2013). Revealing the criterion on botnet detection technique. *International Journal of Computer Science Issues (IJCSI), 10(2), 208.* Retrieved https://www.proquest.com/openview/a8171b8b0d6abc939ef2f886729a5299/1?pq-origsite=gscholar&cbl=55228

Tuan, T. A., Long, H. V., Son, L. H., Kumar, R., Priyadarshini, I., & Son, N. T. K. (2019). Performance evaluation of botnet distributed denial of service attack detection using machine learning. *Evolutionary Intelligence*, *13*(2), 283-294.doi:10.1007/s12065-019-00310-w

Van Roosmalen, J., Vranken, H., & Van Eekelen, M. (2018, April). Applying deep learning on packet flows for botnet detection. *InProceedings of the 33rd Annual Symposium on Applied Computing* (pp. 1629-1636).Pau, France: ACM

Wei, J., & Mendis, G. J. (2016, April). A deep learning-based cyber-physical strategy to mitigate false data injection attack in smart grids. *InJoint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG)* (pp. 1-6). Vienna, Austria: IEEE

Lv, Y., Duan, Y., Kang, W., & Li, Z. (2015).Traffic flow prediction with big data: A deep learning approach. *Transactions On Intelligent Transporation Systems,* 865-873. doi: 10.1109/TITS.2014.2345663

Zeidanloo, H. R., & Manaf, A. B. A. (2010). Botnet detection by monitoring similar communication patterns. *Arxiv Preprint Arxiv*, 1004.1232. Retrieved from https://arxiv.org/abs/1004.1232

Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. *Future Generation Computer Systems*, *28*(3), 583-592. doi: 10.1016/j.future.2010.12.006