International Journal of Scientific Research in Computer Science and Engineering
Vol.**12**, Issue.**4**, pp.**77-93**, August **2024**
E-ISSN: 2320-7639
Available online at: www.isroset.org

Research Article

Identifying Botnets within the Traffic Generated By a Network in Two Different Datasets

Grace Bunmi Akintola[1*]

[1]Dept. of Cyber Security, Nigerian Defence Academy, Kaduna, Nigeria

**Corresponding Author: gbogundele@nda.edu.ng*

Received: 24/Jun/2024; **Accepted**: 26/Jul/2024; **Published**: 31/Aug/2024

Abstract— The impact of cyber-attacks on organizational and private networks has been significant, causing extensive damage and posing serious threats to cybersecurity. This is largely due to the increasing sophistication of malicious hackers, making the detection and mitigation of these attacks more challenging. One such attack is the botnet attack, which involves using compromised systems to launch attacks, including Denial of Service (DoS) attacks, against victim systems. As a result, comprehensive literature reviews have been conducted to examine existing botnet defense and detection techniques, with a particular focus on machine learning due to its effectiveness in identifying and classifying botnet attacks within networks. This paper presents the development of an Artificial Neural Network (ANN) model, a supervised machine learning technique, using MATLAB software for creating, training, and simulating networks. Two datasets, KDD CUP'99 and UNSW-NB15, were used to demonstrate the effectiveness of the proposed model by extracting the same set of features from both. The model achieved classification accuracies of 99.88% and 96% for the respective datasets. A confusion matrix plot was used to illustrate these accuracy values in detail, further validating the model's effectiveness by showing very low false negative and false positive rates in identifying and grouping botnet attacks.

Keywords— Botnets, Networks, Machine Learning, MATLAB, DoS attacks, detection techniques, and datasets

1. Introduction

In the rapidly advancing era of Information Technology, cyber-attacks have become a significant challenge for both users and organizations. Over the years, malicious hackers have carried out these attacks using various techniques and methods, driven by motives such as fun, show of skills, revenge, or political benefits. [1]. Botnet attacks happen to be one of the emerging cyber-attacks which drastically increase due new skills gained by the malicious attacker to launch in such a way that the detection of the attacks within the network will be difficult. Botnet attacks can be described as types of cyber-attacks that involve a bot master or Master handler using already compromised systems that are networked together and they are referred to as botnets or zombies. These exposed systems are then used to carryout further attacks such as DoS, and DDoS, and phishing attacks against the victim system (target). [2].

Denial of Service (DoS) attacks are a type of cyber-attack that disrupts network availability for legitimate users. In this attack, a bot controller uses the already attacked systems to overwhelm a target system by sending a massive volume of malicious packets. This excessive traffic exceeds the target system's capacity, causing it to crash and become inaccessible. [3]. Many legitimate users have encountered this issue when trying to navigate the internet, preventing them from utilizing the resources they requested from a server. Distributed Denial of Service (DDoS) attacks occur when an attacker, or bot controller, uses multiple compromised systems to launch attacks against several targets simultaneously. These compromised systems, which can be remotely controlled, are used to carry out large-scale attacks on victim systems. This type of cyber-attack has remained a constant threat on the internet.. [4].

DDoS attacks are carried out through the use of botnets, which are controlled by a botmaster through a command and control (C&C) channel. [5]. To recognise botnet attacks and curtail their impact on networks, researchers have proposed various techniques for identifying these attacks within network traffic. Among these, machine learning has emerged as a prominent approach. This technique involves using available datasets along with models and algorithms to train the system, enabling it to achieve accurate results that can inform decision-making [6]. This approach functions similarly to the human brain, where outputs are generated based on the input data provided to the selected model.

Machine learning techniques are categorized into three types: supervised, unsupervised, and reinforcement learning. Supervised machine learning produces outputs based on a provided labeled dataset (known data). It utilizes algorithms such as random forests, decision trees, and support vector machines. [7]. Reinforcement learning involves learning from new situations through a trial-and-error process, much like how humans learn from their experiences, allowing the system to continually improve itself [8]. Among these three types of learning, supervised machine learning is often the easiest to use. It simplifies learning from the provided data features to make predictions and address complex relationships and issues between the input and output layers, making it an effective approach [9].

Researchers have adopted several machine learning techniques for various activities, including anomaly detection and signature-based methods to identify anomalous packet behaviour within networks. A hybrid approach was proposed for classifying malware integrating expert-defined features with those learned through deep learning from raw data. This method utilizes deep learning to extract features such as N-grams from assembly instructions and malware bytes, texture patterns, shapelet-based features from grayscale images, and structural entropy. These deep features are then combined with hand-crafted features using a gradient-boosting model through an early-fusion mechanism [10]. Machine learning algorithms and blockchain have been one of the techniques used in generating accurate results (outputs) from large datasets hence helping in predicting and identifying vulnerabilities in IoT-based systems. Blockchain technology helps in providing secure and well-cleared decentralized record-keeping, while machine learning algorithms can evaluate large volumes of data to generate valuable perceptions. The combination of these techniques has the ability to transform industries as they help enhance efficiency via automated and trustworthy processes, enable data-driven decision-making, and strengthen security by minimizing vulnerabilities and guaranteeing information integrity [11].

Building on recent research into numerous techniques for detecting botnets within network traffic, this paper proposes adopting a supervised machine learning approach. We will use the KDD CUP'99 and UNSW-NB15 datasets to train and analyze the data, focusing on detecting botnets, specifically DDoS attacks, from normal traffic based on selected features (input data). The MATLAB tool was employed for analyzing and training the dataset using the Artificial Neural Network (ANN) model, and evaluating its performance. This paper specifically applies the ANN model to identify and analyze DoS attacks, a class of attacks present in the KDD CUP'99 and UNSW-NB15 datasets. This research paper is organized into several sections as follows: Section 1 Introduces cyber-attacks, with a particular focus on botnet attacks and their operational mechanisms within networks. Section 2 Reviews related work on anomaly detection mechanisms, machine learning techniques, and Intrusion Detection Systems (IDS) for identifying anomalous behavior within networks.
Section 3 describes the research methodology, including the justification for the chosen methods, details of the datasets used, their collection and extraction processes, the analysis tools selected, and the rationale for the proposed model. Section 4 outlines the experimental design, including the processing of training and testing data for both the KDD CUP'99 and UNSW_NB15 datasets, the generation and selection of thresholds for computing accuracy values, and the use of confusion matrix tools to validate performance and section 5 Concludes the research and offers recommendations for future work.

2. Related Work

This section reviews various academic journals and conference papers focused on existing detection techniques and frameworks for identifying and classifying botnet attacks within networks. The methods discussed are categorized into anomaly-detection mechanisms within the network, machine learning techniques, and Intrusion Detection System mechanism for Anomaly Behaviour within Network.

2.1 Several anomaly-detection mechanisms within the network (methods)

A study was conducted on noninvasive inspection methods to offer a comprehensive overview of recent advancements in anomaly detection. This study reviewed research from the past five years, focusing on new technologies and future opportunities in this field. The literature review specifically addressed anomaly detection systems within network traffic, including applications in Wireless Sensor Networks (WSN), the Internet of Things (IoT), High-Performance Computing, Industrial Control Systems (ICS), and Software-Defined Networking (SDN) environments. The review also highlights various pending issues that must be tackled to enhance anomaly detection systems. [12].

Another research conducted a network traffic anomaly detection model that deals with addressing the issues of high-dimensional abnormal traffic and overfitting due to outliers. The model calculates mutual information to choose optimal features and employs a chaotic neural network algorithm with an adaptive strategy to refine feature selection. This approach enhances data quality and significantly improves classification performance and prediction accuracy, reducing training time by over 12% without compromising the original model's performance metrics [13].

A novel online anomaly detection system based on Software-Defined Networks (SDN) was developed and this system utilizes a Convolutional Neural Network (CNN) to promptly excerpt and analyze original network flow features, allowing for real-time packet extraction and recognition. Utilizing SDN allows the system to flexibly adapt to network changes, resulting in a zero-configuration anomaly detection system. The system's packet filter automatically implements mitigation strategies, achieving real-time mitigation of abnormal traffic. Experimental results demonstrate that this system is highly accurate, promptly alerts network managers to enable timely security measures, and effectively detects abnormal traffic, thereby enhancing the security performance of edge clustering networks [14].

A network encryption traffic classification model was introduced that combines attention mechanisms with spatiotemporal features. The model adopts LSTM (long short-term memory) to analyze temporal correlations in continuous network flows, CNN (Convolutional Neural Network) to extract high-order spatial features, and the SE (Squeeze and Excitation) module to weigh and redistribute these features. This three-stage process enables fast classification of network flows. Key advantages include automatic mapping of network flow to labels without manual intervention, strong generalization to various datasets, and high accuracy in handling encrypted applications and their traffic. Experimental results show the model achieves over 90% accuracy in classifying encrypted and unencrypted network traffic [15].

A systematic review was conducted for the AI-based anomaly identification techniques for encrypted traffic. Several research questions were formulated and studies were based on specific eligibility criteria. After performing the vetting process and quality assessment, 30 highly relevant research articles were chosen for inclusion. These studies were reviewed focusing on datasets, feature extraction, feature selection, preprocessing, anomaly detection algorithms, and performance indicators. The review confirmed that a variety of techniques are employed for AI-based anomaly identification over encrypted traffic. While some methods are identical to those adopted for unencrypted traffic, others are distinct and specific to encrypted traffic [16].

Another study was developed which involves proposing a new deep-learning-based traffic anomaly detection model by enhancing feature-engineering methods, aiming to improve efficiency and accuracy. The research comprised two main aspects namely: Dataset Construction and Data Detection Algorithm Model. The dataset construction involves the use of the UNSW-NB15 dataset, the study integrates feature extraction standards and methods from other datasets to create a comprehensive feature description set, and the resulting dataset, DNTAD, demonstrated improved operational efficiency and maintained the training performance of machine learning algorithms like XGBoost. The Detection Algorithm Model introduces an LSTM-based detection algorithm with a self-attention mechanism to capture important time-series information in traffic data. The model effectively learns time dependencies and the relationships between traffic features, confirmed through ablation experiments. The proposed model performs better than other comparative models on the refined dataset [17].

A thorough analysis was performed on two primary approaches in network traffic anomaly detection: feature recognition and anomaly detection. Each approach expresses unique strengths and faces particular issues. The study explores how integrating deep learning with artificial immune systems could likely revolutionize feature identification. Additionally, it demonstrates improvements in anomaly detection by combining machine learning techniques with traditional methods. Looking forward, the paper outlines research directions focused on integrating deep learning, artificial intelligence, and behavioral analysis. This integration works on enhancing network traffic anomaly monitoring systems' precision, efficiency, and adaptability. Proposed future strategies include advancements in data preprocessing, model development, pattern recognition, and adaptive adjustments, all aimed at strengthening network defenses against the evolving landscape of cyber threats [18].

2.2 Various Machine Learning Techniques

The challenge of anomaly detection in network traffic was tackled by proposing a three-stage framework that operates solely on normal traffic data. The approach generates pseudo-anomaly samples without prior anomaly knowledge to facilitate anomaly detection. The first process was the use of a reconstruction method to learn deep representations of normal samples and the second process involves the representations that are normalized to a standard normal distribution through the use of a bidirectional flow module. To simulate anomaly samples, noise is added to these normalized representations, which are then processed through the generation direction of the bidirectional flow module. Finally, a simple classifier is trained to distinguish between normal samples and pseudo anomaly samples in the latent space. During inference, the designed framework relies on just two modules for detecting anomalous samples, significantly reducing model complexity. Experimental results reveal that the method achieves state-of-the-art performance on common benchmark datasets for anomaly network traffic identification [19].

A survey paper was done for examining the security challenges within NFV (Network Function Virtualization) and advocates for the adoption of anomaly recognition techniques to alleviate cyber-attack risks. The research analyses machine learning-based algorithms' strengths and weaknesses in detecting network anomalies specific to NFV networks. By identifying the most effective algorithms for timely and accurate anomaly detection, this study aims to empower network administrators and security professionals in bolstering the security of NFV deployments. Ultimately, this effort aims to protect the integrity and performance of sensors and IoT systems in NFV environments [20].

The CTU-13 dataset, which is a widely used resource in cybersecurity was adopted to develop a machine learning-based method for recognizing botnets. The CTU-13 dataset comprises real network traffic data recorded in an environment compromised by a botnet. Various machine learning algorithms, including decision trees, regression models, naïve Bayes, and neural networks, are trained to classify network traffic as either botnet-related or benign. The performance of each model is determined using criteria such as accuracy, precision, and sensitivity, measuring their effectiveness in identifying both known and unknown botnet traffic patterns. Experimental results demonstrate the high accuracy of this machine learning approach in detecting botnets, with impressive detection rates and low false positive rates, indicating its potential for real-world application [21].

Another study was explored about the application of various machine learning methods, including logistic regression,

random forest, and deep neural networks, for enhanced classification performance. Deep learning algorithms were applied to artificial neural networks (ANN) in multiple ways. To achieve more accurate results compared to traditional learning methods, datasets were split into two halves and underwent specialized pre-processing. This pre-processing aimed to enhance the performance of the classification algorithms. The results from the deep learning approach were promising, achieving an accuracy of 99.79% for SMS attacks and 98.48% for malware attacks [22].

In ensuring the robustness, reliability, and security of a system, mechanisms for botnet detection and removal are intensively reviewed. Botnets are typically grouped based on the protocol used by their command-and-control servers, such as IRC, HTTP, DNS, or Peer-to-Peer (P2P). Detection of botnets can be achieved using various algorithms, including decision trees, random forests, K-nearest neighbors (KNN), naïve Bayes, and support vector machines (SVM). The research analyzed various studies on botnet attacks and detection techniques, providing insights into the effectiveness of different methods [23].

Analysis of diverse machine learning algorithms for botnet detection was carried out. Multiple machine learning algorithms are adopted and their effectiveness in detecting botnets is evaluated. Using an existing dataset, the algorithms are tested and the results demonstrate their capability in accurately identifying botnet activity [24].

The thesis was carried out to tackle the anomaly detection complications in edge cloud environments. By exploring various anomaly detection strategies and leveraging machine learning techniques, this research seeks to enhance the efficiency and accuracy of detecting anomalies in such environments. The proposed methods aim to overcome challenges such as resource limitations, the lack of labeled data specific to edge clouds, and the need for accurate anomaly detection. Emphasizing on machine learning techniques including transfer learning, knowledge distillation, reinforcement learning, deep sequential models, and deep ensemble learning, this thesis attempts to establish coherent and accurate anomaly detection systems tailored for edge cloud environments. The results demonstrate significant improvements achieved by employing machine learning methods for anomaly detection in edge clouds. Extensive testing and evaluation in real-world edge environments reveal that machine learning-driven anomaly detection systems improve the recognition of anomalies in edge clouds. These methods achieve a reasonable trade-off between accuracy and computational efficiency. The discoveries clearly display how machine learning-based anomaly detection approaches contribute to building resilient and secure edge-based systems [25].

A study was conducted which involves examining papers published between 2015 and 2023 that focus on anomaly detection using machine learning techniques. After evaluating the selected research papers, Ten (10) diverse applications of anomaly detection were identified and outlined in the publications. It was realized that machine learning models were used to detect anomalies in 6% of all instances. Additionally, a comprehensive list of datasets was conducted that was used in detecting anomaly, including many other general-purpose datasets. Also, the analysis revealed that, compared to other categorized anomaly detection methods, researchers are more inclined to employ unsupervised anomaly detection techniques. The application of machine learning models for anomaly detection is one of the most promising fields of study, with researchers utilizing various ML models in this context. Based on the results of this review, recommendations and suggestions were offered to researchers in the field [26].

A systematic literature review was conducted to provide an overview of deployed models, data pre-processing mechanisms, anomaly detection techniques, and their evaluations. Instead of quantitatively comparing existing approaches, this survey works on helping readers understand important aspects of different model architectures and highlight open issues for future research [27].

Finally, research was proposed for the botnet identification system, ACLR, stacking artificial neural network (ANN), convolutional neural network (CNN), long short-term memory (LSTM), and recurrent neural network (RNN). Experiments compare individual models with ACLR using the UNSW-NB15 dataset, encompassing nine attack types. ACLR achieves 0.9698 testing accuracy, effectively capturing botnet attack patterns. K-fold cross-validation at $k = 5$ shows ACLR's robustness (accuracy 0.9749). ACLR detects botnets with ROC-AUC 0.9934 and PR-AUC 0.9950. Comparative analysis with state-of-the-art models confirms ACLR's superior performance, enhancing cybersecurity against evolving threats [28].

2.3 Various Intrusion Detection System mechanism for Anomaly Behaviour Within Network

A framework called IoTBoT-IDS was proposed for statistical learning-based botnet recognition to enhance the security of IoT-based smart networks against botnet attacks. This framework captures and detects the normal characteristics of IoT networks using the Beta Mixture Model (BMM) and the Correntropy Model. Deviations from normal traffic patterns are categorized as abnormal or malicious. Evaluation results showed that the proposed framework achieved an average detection accuracy of 99.2% [29].

The Intrusion Detection Dataset Toolkit (ID2T) is another technique designed to address the challenges of recreating datasets with selected features, aiming to produce accurate scientific results. The ID2T architectures facilitate the injection of latest and advanced attacks, enabling precise detection of potential abnormalities in network traffic [30].

A paper was conducted as regards introducing a security system named Detection of Anomalous Behaviour in Smart Conveyance Operations (DAMASCO) which was designed for intrusion detection in vehicle-to-vehicle (V2V) communication. Adopting a statistical approach, the anomaly

detection module targets the Medium Access Control (MAC) sublayer to monitor the number of packets sent, detect potentially harmful nodes, obstruct their activity, and sustain a reputation list. The algorithm employs the Median Absolute Deviation (MAD) to detect outliers and characteristics of DoS attacks. Experiments conducted in a simulated environment with a realistic urban mobility model demonstrate that the introduced system achieves a 3% false positive rate and no false negatives [31].

A hybrid feature selection strategy was introduced to improve network anomaly and illegal traffic detection. By integrating linear-based and tree-based feature selection methods, the approach creates a robust feature representation that improves the accuracy and efficiency of intrusion detection systems (IDS). The study adopts a supervised machine learning approach using decision trees and employs 10-fold cross-validation for rigorous evaluation. Experimental results across diverse datasets consistently show accuracy levels exceeding 99%. Moreover, the proposed strategy achieves a weighty reduction in the number of features, up to 78%, for enhancing detection accuracy. This reduction facilitates simpler models, substantial resource savings, and faster response times, specifically advantageous for real-time intrusion detection systems [32].

3. Research Methodology

After reviewing numerous research papers on various detection techniques proposed by researchers, it has been found that there are insufficient defense and detection methods to effectively mitigate the impact of DoS (Denial of Service) and DDoS (Distributed Denial of Service) attacks within networks.. Therefore, this section comprised of the justications of theadopted research method, the selected datasets, their collection and extraction processes, the selcetd analysis tool used, and Justification of the proposed Model.

3.1 Justification of Adopted Research Method

A quantitative research methodology was chosen for this study as it involves the design and simulation of a network model, which is based on analyzing and measuring performances numerically (in graphical form). This methodology was selected due to its ease of interpreting the obtained data (results) and its high level of accuracy, as assessed via the generated thresholds.

3.2 Justification of the used Datasets

To achieve the aim of this paper, two different datasets were collected and analyzed namely: KDD CUP'99 and UNSW_NB15 datasets. KDD (Knowledge Discovery in Databases) CUP'99 dataset was selected due to its systematic method to revealing, refining, and utilizing meaningful insights and patterns within raw databases for various applications. This methodical process of data exploration, transformation, and refinement extracts actionable knowledge. The advantages of the KDD process in data mining are extensive, including informed decision-making, improved business performance, enhanced efficiency, better customer experiences, fraud detection, and predictive modeling. By embracing KDD, organizations can access the full usage of their data, driving innovation, growth, and success in today's data-driven landscape [33]. While UNSW_NB15 dataset is viewed as a indepth dataset mainly for network intrusion detection systems. It encompasses nine different types of attacks, such as DoS, worms, backdoors, and fuzzers. The dataset includes raw network packets, capturing a wide range of modern network intrusion scenarios, covering both different types of attacks and normal traffic [34].

3.3 Justification of the selected analysis tool

In achieving the aim and objectives of this paper, MATLAB 2021a software was used for the simulation and analysis of the collected datasets (KDD CUP'99 and UNSW_NB15). MATLAB offers a range of functionalities, including efficient numerical computation and a vast collection of built-in functions and toolboxes across various domains such as signal processing, image processing, control systems, optimization, and machine learning. It provides comprehensive tools for data visualization and plotting, with rich capabilities for creating diverse 2D and 3D plots. Additionally, MATLAB provides a maximal level of interaction with programming languages like C/C++, Java, Python, and .NET, allowing users to exploiting funtioning code and libraries from different languages, thereby enhancing MATLAB's capabilities and extending its reach to incorporate external functionalities [35]. The MATLAB software was installed on a Windows 10 Pro operating system with 4.00 GB of RAM and an Intel® Pentium® CPU 2020M @ 2.50 GHz.

3.4 Dataset collection and extraction process

The KDD CUP'99 dataset was downloaded from datahub.io, and the UNSW_NB15 dataset was obtained from CloudStor (aarnet.edu.au). PyCharm software was then used to view the dataset information, extract the necessary features, and save them as CSV files in a project folder before importing them into Microsoft Excel. In Microsoft Excel, the data was processed into binary format (0s and 1s). The dataset included records of Smurf, Neptune, and Back (types of DoS attacks) as well as normal traffic. To prepare the data for analysis, 70% of the records, covering all DoS attack types and normal traffic, were allocated for training, while 30% were set aside for testing.

Lastly, the datasets were imported into MATLAB for analysis to identify botnet attacks (specifically DoS attacks) and distinguish them from normal traffic. The same processing steps were applied to the UNSW_NB15 dataset, dividing it into training and testing data.

Out of the whole features included in KDD CUP'99 and UNSW_NB15datasets, four main features were extracted separately from each dataset. This was done after the dataset had been downloaded and then imported into pycharm Professional 2021.2.1 software. Using pycharm helped in completing smart codes as it supports writing Python, javascript, CSS, and other languages which helps in easily detecting errors(highly sensitive to any type of code), code

editing, and fixing the errors when running. It can as well be adopted to any related work that includes importing, creating, editing, and saving files.

Figure 1 explains the set of Python codes used for easy view of the whole KDD CUP'99 dataset and UNSW_NB15 dataset as this seems better compared to using Microsoft Excel due to the large volume of data (including the columns). The python code was used for the extraction of features in both datasets. The screenshot shows the overall information of the dataset such as the the headers (features) and the total number of data records. Python code makes it easier to extract the chosen features without having to scroll through the entire data. A total of 42 features were identified from the KDD CUP'99 dataset, including the label feature that classifies each data record by attack class type and specifies their data types. Among these features, four were selected for their critical importance in understanding packet transmission within a network: source bytes, destination bytes, duration, and counts.

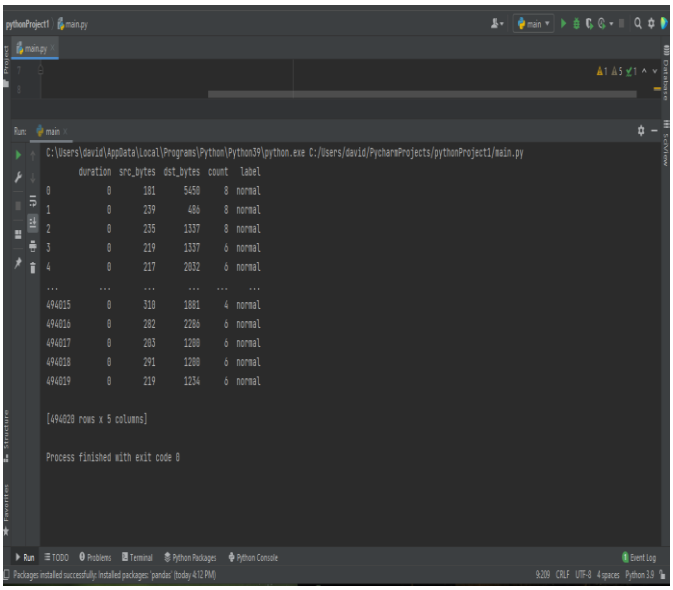
These features are integral to the packet header and provide essential details about packet contents, source, and destination. Specifically:

I. **Source bytes** indicate the number of bytes sent from the source IP address.
II. **Destination bytes** represent the number of bytes received by the destination IP address during data transmission.
III. **Duration** reflects the time (in seconds) taken for the packet to travel from the source IP address to the destination IP address.
IV. **Count**s reveal the number of connections from the source and destination IP addresses, compared to connections established in the past two seconds.

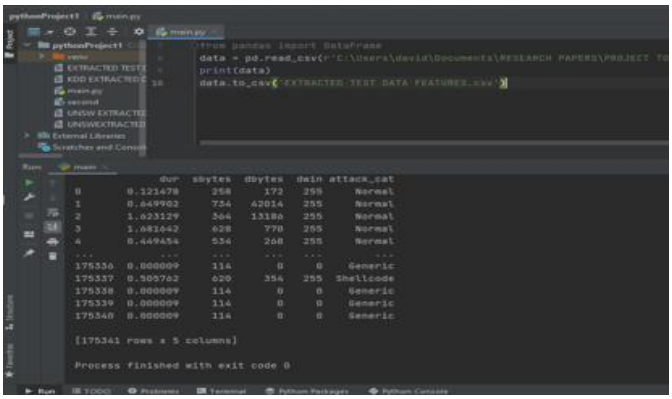
These features are crucial for analyzing network traffic and understanding packet flow. [36]. The same four features extracted from the KDD CUP'99 dataset—duration, source bytes, destination bytes, and count—were also extracted from the UNSW_NB15 dataset. In the UNSW_NB15 dataset, the count feature is labeled as "dwin."

i. **Duration** is defined as the time difference between the first and last packet of a network connection, measured in seconds
ii. **Source bytes** represent the number of data bytes sent from the source to the destination.
iii. **Destination bytes** are the number of data bytes received by the source from the destination.
iv. **Count** (dwin in UNSW_NB15) indicates the number of packets counted within the specified packet group.

These features were selected to ensure consistency in classifying and detecting DoS attacks, a type of botnet attack. [37].



Extraction of KDD'CUP'99 data features



Extraction of UNSW_NB15 data features

Figure 1: features extraction of both datasets

3.5 The Proposed Model Justification

The proposed architecture utilizes a neural network model to detect and analyze botnet attacks. Botnet attacks are orchestrated by botmasters who use compromised systems (bots) to launch DoS (Denial of Service) or DDoS (Distributed Denial of Service) attacks. The model is designed to recognize and classify traffic patterns associated with DoS attacks [38]. The application of an Artificial Neural Network (ANN) model is essential for reducing the false positive rate in detecting botnet attacks, thereby improving the accuracy of distinguishing normal traffic from botnet (DoS) attack traffic [39].

For the KDD CUP'99 dataset, the model focuses on three types of DoS attacks: Smurf, Neptune, and Back. In the UNSW_NB15 dataset, only the DoS attack class and normal traffic are considered, in alignment with the paper's emphasis on botnet attacks.

a) **Smurf Attacks:** These are a type of DDoS attack that floods the target system with packets by exploiting the Internet Control Message Protocol (ICMP). This involves creating large spoofed packets with a fake source address, which leads to overwhelming the target system and rendering it slow or inoperable [40].

b) **Neptune Attacks:** Also known as half-open TCP SYN attacks, these involve sending repeated SYN packets with fake IP addresses and random ports to every port on the targeted server. This causes network saturation and results in a high number of SYN error connections compared to other attack types [41].
c) **Back Attacks:** These attacks target Apache web servers by overwhelming them with requests that contain large numbers of forward slashes (/) in the URL, causing the server to become unresponsive. As a result, the server becomes unable to process requests from authorized users (clients) due to the overwhelming volume of incoming requests. This overload forces the server to attempt processing all requests simultaneously, effectively denying service to legitimate users [42].

An Artificial Neural Network (ANN) is described as a supervised machine learning technique where input nodes are processed through hidden layers, with weights applied to generate the expected outputs at the output layer. In this paper, the ANN model utilizes a Multilayer Perceptron (MLP) network, which produces accurate results by learning a general rule that maps inputs to outputs. This approach enhances the effectiveness of the system, making it more reliable compared to other methods. [43]. Multilayer Perceptron (MLP) architecture is one of the most common feed-forward neural network models which effectively identifies basic patterns (behaviors) within neural models. In MLP, the propagation of an impulse occurs in one direction—from the input layer, consisting of four input nodes, through the hidden layer, also known as the "network brain." Here, the sum of weights associated with all neurons is combined, excluding the input bias, to produce the output layer with two output features [43].

Figure 2 illustrates an MLP with a single hidden layer, featuring a decision boundary that surrounds a single convex region of the input space. The process begins at the input layer, where the data vector is accepted. The hidden layer receives output from the input layer, applies weights, and passes the data through a non-linear activation function. Finally, the output layer accepts the weighted outputs from the hidden layer, passes them through an output non-linearity, and generates the target values.

Each circular node in Figure 2 represents an artificial neuron, while the arrows indicate connections between neurons from one layer to the next. These connections form the Artificial Neural Network (ANN), enabling neurons to signal one another as the information is processed. In the ANN model diagram (Figure 2), the input features are represented as (X_i), where i = (X_1, X_2, X_3, $X_{4,\ldots}n$) indicates the number of input features extracted from the dataset for model training. The hidden layer is denoted as (H_n), where (n = 1, 2, 3, 4,….n) represents the number of nodes in the hidden layer. Lastly, (Y) represents the outputs from the hidden layer after being weighted (W_{in}) from both the input and hidden layers, which are then passed to the output layer as $Y_1$and Y_2 (the two output features).

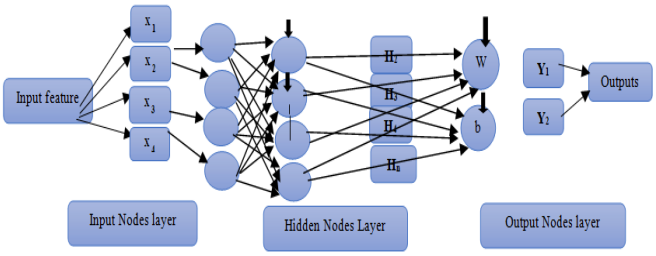


Figure 2: ANN Model with Multilayer perceptron

The linear activation function (P) is implemented to both the hidden and output layers, while (α) represents the bias, and (W_{in}) represents the weights for individual linked neurons. This process involves multiplying the inputs (X_i) by the weight ((W_{in}), adding the constant bias (α), and applying the activation function to produce the final outputs.

Figure 3 explains the basic steps involved in applying the Artificial Neural Network model The first step involves the collection of KDDS CUP'99 dataset which are DoS attack class types, these include Smurf, Neptune and Back attacks as well as the normal traffic type. These same steps were repeated for the processing of UNSW_NB15 dataset.

Secondly, the dataset was processed, and four (4) features are chosen from the DoS attack class types that were selected the normal traffic type as well. Next, the training data and the testing data was be imported into MATLAB for training and analysis. The input and output data for the training and testing datasets was viewed in MATLAB using (typing) a set of commands in the command window.

Next to this, involves the transposition of input data and output data for both training data and testing data, then, nntool (neural network tool) was opened where the transposed features which are the inputs and output of both training datasets and testing datasets was be imported, thus leading to creating network where network type, transposed features (input and output), number of neurons to be applied was be selected for the creation of the outputs of the network.

Both output and input features needed to be transposed in order to fit into the designed model, hence generating accurate results. After this, the transposed features (input and output) of both training data and training data was trained in order to display number of iterations conducted as well as the performance graph which determine its accuracy.

The next step was to simulate only the input transpose feature (only input transpose feature will be chosen) producing Simulated network outputs which were compared to the actual output (output transpose features), if peradventure sufficient accurate results are not gotten, then threshold was obtained, and this was used in training test data to generate an accurate and effective classification of the attacks and distinguishing them from the normal traffic.

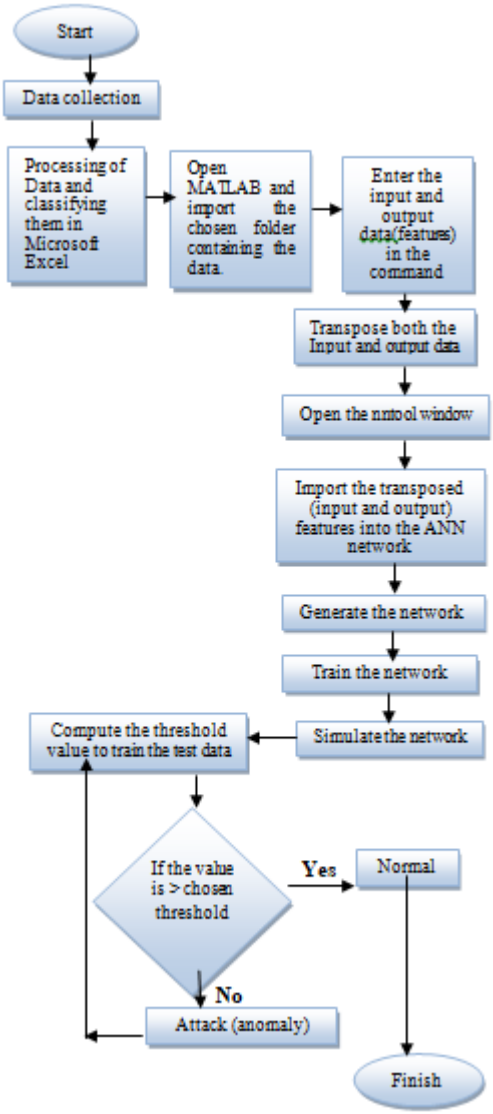


Figure 3: procedures involve in analyzing the collected data

4. Results and Discussion

4.1. Experimental Design Details

The Artificial Neural Network (ANN), also referred to as the Neural Network model, is utilized to analyze the KDD CUP'99 and UNSW-NB15 datasets for accurate detection and differentiation of anomalous traffic from normal traffic. The ANN was chosen for its superior ability to model complex patterns and generate precise predictions. Its strength lies in handling non-linear and intricate relationships between the input and output layers, transforming inputs into meaningful outputs through its activation function [44]. When applying the neural network model for data analysis, the Levenberg-Marquardt algorithm is considered among other algorithms used for training neural networks. This algorithm aims to minimize the sum of squared errors between the model function and the data points through a series of carefully selected updates to the model parameters. Specifically designed to address the sum of squared errors, the Levenberg-Marquardt algorithm utilizes loss functions and operates with the gradient vector and the Jacobian matrix. The Jacobian matrix, constructed from the first-order partial derivatives of scalar functions relative to a set of independent variables, is primarily used for analyzing small signal stability within the system [45]. The Levenberg-Marquardt backpropagation algorithm (trainlm) is specifically used for training both the training and test data. This algorithm functions as a network training tool that updates weight and bias values based on Levenberg-Marquardt optimization. It is considered the fastest backpropagation method and is recommended as the first-choice supervised algorithm, despite its higher memory consumption compared to other algorithms [45]. The experimental neural network, designed to simulate datasets for the effective detection of anomalous (malicious) traffic, was developed using MATLAB 2021a.

Understanding the information provided by each header (which is the feature) of the dataset highlights their significanees in detecting botnet attacks. These attacks aim to use compromised systems (bots) to launch assaults on target systems, ultimately rendering services and resources unavailable to clients on the internet [46]. The selected features are instrumental in determining the number of bytes transferred between the source and destination IP addresses, and in detecting delays between these endpoints. This information is crucial for identifying delays or complete unavailability of resources for clients (victim systems). As illustrated in Figure 4, the input layer consists of these four selected features, which are processed through the hidden layer. This hidden layer, equipped with ten neurons, utilizes weights (w) and biases (b) to achieve optimal attack classification results. The TANSIG (Hyperbolic Tangent Sigmoid) function is applied in the hidden layer to produce effective results. TANSIG is a nonlinear activation function used in artificial neural networks to compute the output of a layer based on its net input. [47]. The output layer portrays the network traffic (output) features in binary form, with two outputs indicated as 0s and 1s. This layer employs the PURELIN function, a linear activation function that computes the layer's output based on its net input. [48].

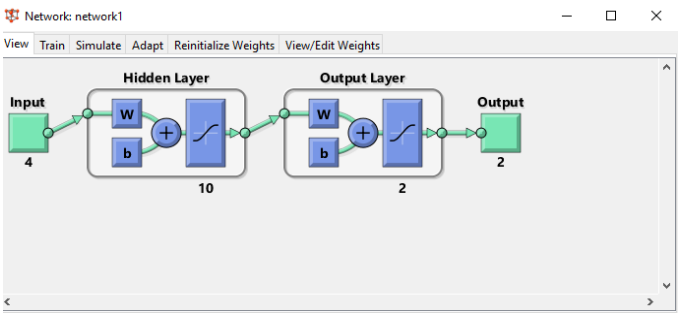


Figure 4: The generated Artificial Neural Network Model

4.2. Extracted and processed features of the collected datasets

The KDD CUP'99 dataset includes various attack types, but this paper focuses on just two: normal traffic and DoS (Denial of Service) attacks. For the UNSW_NB15 dataset, which does not have subclasses for DoS attacks like the KDD dataset, the analysis also concentrates on DoS attacks and normal traffic. From a total of 3,000 extracted records, 2,100

(comprising both DoS attacks and normal traffic) were designated for training, while 900 records (also a mix of DoS attacks and normal traffic) were used for testing. Three specific DoS attack types were selected for analysis: Back, Neptune, and Smurf attacks. In total, 6,000 records of attack traffic were used, with 70% of records from each attack type allocated to the training dataset and 30% to the testing dataset. Output features were represented as Attack Bit1 and Attack Bit2, encoded in binary format (0s and 1s). As outlined in Table 1, the datasets consist of selected attack types: the KDD CUP'99 dataset includes Back, Neptune, and Smurf DoS attacks along with normal traffic, while the UNSW_NB15 dataset includes only DoS attacks (without subclasses) and normal traffic. Both datasets feature four input attributes (Duration, Source Bytes, Destination Bytes, and Counts) and two output attributes (Attack Bit1 and Attack Bit2), represented as binary values (0s and 1s) before being imported into MATLAB.

Table 1: Attack Class types selected from KDD CUP'99 and UNSW_NB15 datasets

KDD CUP'99 DATASET			
Attack class types		**Number of features selected**	
DoS Attacks	**Normal Class Type (Normal Traffic)**	**Selected Input Features**	**Generated output Features**
Back		Duration, Source bytes, Destination bytes, and Counts	Attack Bit1 Attack Bit2
Neptune			
Smurf			
UNSW_NB15 DATASET			
Attack class types		**Number of features selected**	
DoS attacks	**Normal Class Type**	**Selected Input Features**	**Generated output Features**
		Duration, Source bytes, Destination bytes, and Counts (named as "dwin")	Attack Bit1 Attack Bit2

Table 2 details the distribution of KDD dataset records across different attack types. For each attack type, 70% of the total records were allocated to the training dataset, while 30% were used for testing. The output features for each attack type are as follows: i. Back Attack: (0, 0), ii. Neptune Attack: (0, 1), iii. Smurf Attack: (1, 0), and iv. Normal Traffic: (1,1)

This resulted in a total of 4,200 training records to be used in MATLAB for network training and output generation. In the UNSW_NB15 dataset, 70% of the data records were designated for training, encompassing both DoS attack types and normal traffic. The output features in this dataset were represented as: i. DoS Attack Type: (0, 1), and ii. Normal Traffic: (1, 0). The total number of training records used from the UNSW_NB15 dataset was 2,100.

Table 2: Computation for Training dataset records IN KDD CUP'99 and UNSW_NB15 datasets

Training dataset records IN KDD CUP'99 dataset estimation			
Back Attacks	**Neptune Attacks**	**Smurf Attacks**	**Normal class**
Total number of records = 1,500 records 70% of 1,500 = 1,050	Total number of records = 1,500 records 70% of 1,500 = 1,050	Total number of records = 1,500 records 70% of 1,500 = 1,050	Total number of records = 1,500 records 70% of 1,500 = 1,050
Total Number of Records = 4,200			
The Training dataset records (UNSW_NB15) estimation			
DoS attacks		**Normal Class**	
Total number of records = 1,500 records 70% of 1,500 = 1,050		Total number of records = 1,500 records 70% of 1,500 = 1,050	
Total Number of Records = 2,100			

Table 3 illustrates the generation of the testing dataset, which comprises 30% of the total records for each attack class type (Back, Neptune, Smurf) and Normal traffic. This equates to 450 records for each class type. The output features are represented as follows: i. Back Attack:(0,0), ii. Neptune Attack: (0,1),iii. Smurf Attack: (1,0), and iv. Normal Traffic:(1,1)

For the UNSW_NB15 dataset, 30% of the total records were allocated to the testing dataset, including DoS attack types and normal traffic. The output features are denoted as i. DoS Attack Type: (0, 1), and Normal Traffic: (1,0). The total number of records used for testing in the UNSW_NB15 dataset was 900

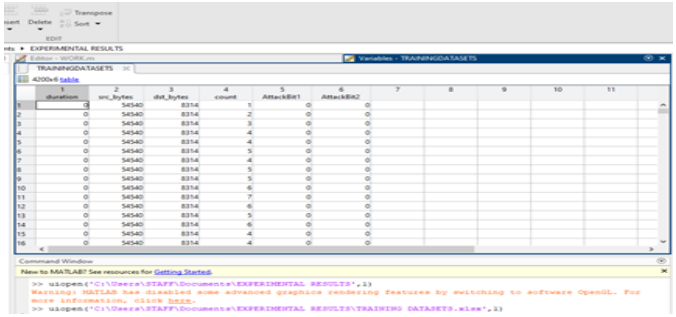
Table 3: Table 3: Computation for Testing dataset records in KDD CUP'99 and UNSW_NB15 datasets

The Testing dataset records (KDD CUP'99) estimation			
Back Attack	**Neptune Attack**	**Smurf Attack**	**Normal Class**
Total number of records = 1500 records 30% of 1,500 = 450	Total number of records = 1500 records 30% of 1,500 = 450	Total number of records = 1500 records 30% of 1,500 = 450	Total number of records = 1,500 records 70% of 1,500 = 450
Total Number of Records = 1,800			
The Testing dataset records (UNSW_NB15) estimation			
DoS attacks		**Normal Class**	
Total number of records = 1500 records		Total number of records = 1500 records	

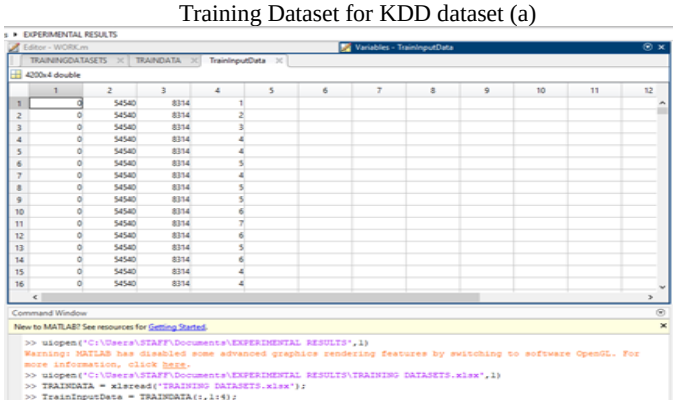
30% of 1,500 = 450	30% of 1,500 = 450
Total Number of Records = 900	

4.3 The processing of KDD CUP'99 Training Data in MATLAB

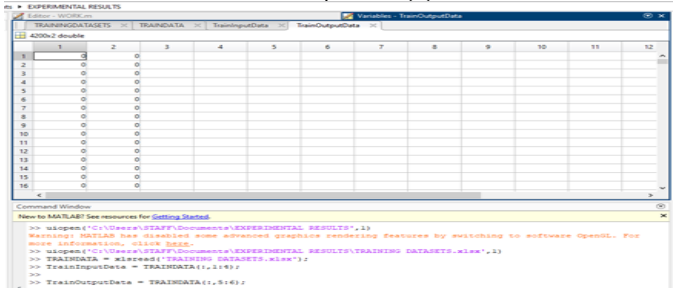
The training data, which constitutes 70% of the entire KDD CUP'99 dataset, was first downloaded and organized in Microsoft Excel. This data was then imported into MATLAB using the command `nntool` in the MATLAB command window, allowing for easy viewing and manipulation within the environment as depicted in Figure 5(a). Next, the training data's output features (which contain four features) were separated from the input features (which include two features) by converting the numerical values to a double data type format and saving it as "TRAINDATA" using a command in the MATLAB command window. The input features, occupying columns 1 to 4, were saved as "TrainInputData" (as shown in Figure 5(b)). Meanwhile, the output features, located in columns 5 and 6, were saved as "TrainOutputData" (as shown in Figure 5(c). Both TrainInputData and TrainOutputData variables were then transposed to be suitable for the designed ANN model, hence getting accurate results for the analysis.



Training Dataset for KDD dataset (a)



TrainInputData (b)

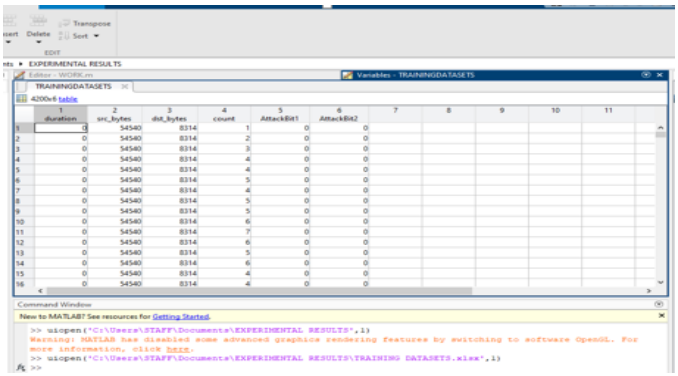


TrainOutputData (c)

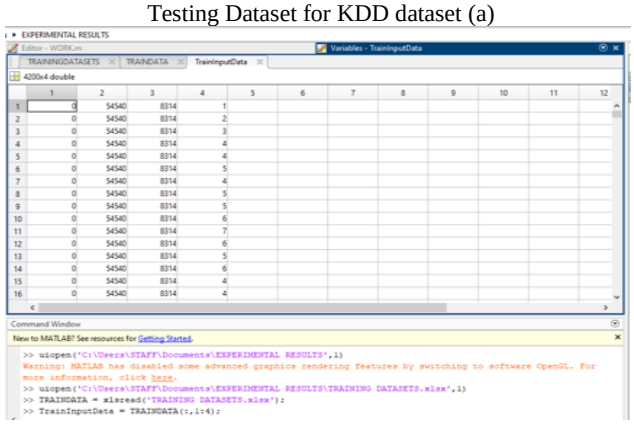
Figure 5: The process of training data for KDD dataset

4.4 The processing of KDD CUP'99 Testing Data in MATLAB

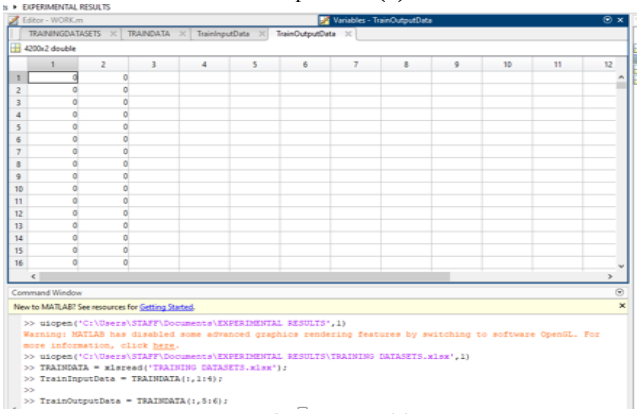
The same process used for the training data was repeated for the testing data, which comprised 30% of the whole dataset (KDD CUP'99). It was imported from Microsoft Excel into MATLAB using the line of code in the command window (nntool) for easy readability and Extraction of input and output features from the test data, including converting the table into a double data type format for use in MATLAB and "TESTDATA" variable is created as depicted in figure 6a. The input features occupied columns 1 to 4 and were saved as "TestInputData" in Figure 6(b)while the output features that occupied columns 5 and 6 were saved as "TestOutputData" in Figure 6(c). Both TestInputData and TestOutputData variables were then transposed to be suitable for the designed ANN model, hence getting accurate results for the analysis.



Testing Dataset for KDD dataset (a)



TestInputData (b)



TestOutputData (c)

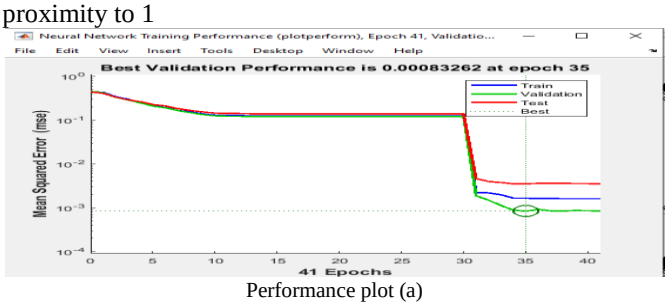
Figure 6: **The testing data process for the KDD CUP'99 dataset**

After transposing both the input and output features of the test data (KDD CUP'99), the neural network was opened using the nntool command where both transposed TestInputData and TestOutputData variables are being imported. The `nntool` in MATLAB is a built-in tool that allows users to open the Network/Data Manager for importing, creating, utilizing, and exporting network outputs[49]. TestInputDataTranspose is imported as input data while TestOutputDataTranspose serves as the network's target data. The transposed data must be imported to create a network for training. As illustrated in Figure 4, the network model includes 4 neurons in the input layer, 10 neurons in the hidden layer using the TANSIG (Hyperbolic Tangent sigmoid transfer) function, and 4 neurons in the output layer where the PURELIN function is used to produce effective network outputs. After creating the network using `nntool`, it is accessed by double-clicking to open the training and simulation page. Here, both transposed `TestInputData` and `TestOutputData` are selected for training the network, and `TestInputDataTranspose` is used for simulating the network. After completing the procedures for creating, importing, transposing features, and training and simulation of networks with the KDD CUP'99 dataset, the same process was applied to the UNSW_NB15 dataset. This involved creating and importing both the training and testing datasets into MATLAB, transposing the input and output features, and importing them into `nntool` (Data Manager) for training and simulation. The resulting network outputs were then evaluated through regression plots, training state plots, and performance plots, as detailed in Section 4.4 for the KDD CUP'99 dataset results

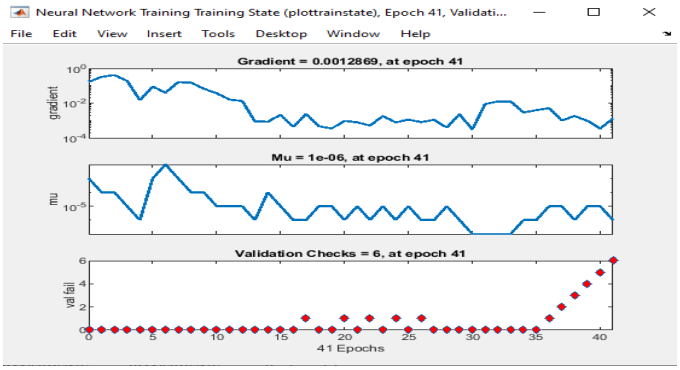
4.5 Interpretation of the graphical plots of Trained data (KDD CUP'99)

Figure 7 displays three key plots from the training results for the dataset:

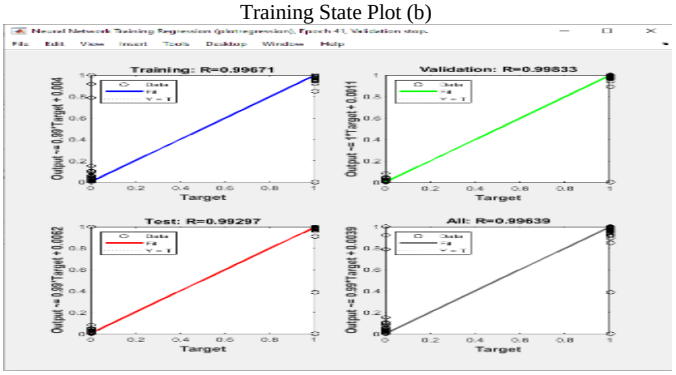
1. **Performance Plot (Figure 7a)**: This plot shows the trained network's Best Validation Performance (BVP), which reached 0.00083262 at epoch 35 out of 41 iterations. The mean squared error is close to zero, indicating effective training. Training and validation both concluded at the 35th epoch.
2. **Training State Plot (Figure 7b):** The gradient result is 0.0012869, the control parameter (MU) is set to 1e-06, and the validation checks total 6, all achieved by epoch 41.
3. **Regression Plot (Figure 7c):** The regression analysis shows that the performance of the trained dataset is 0.99671, the validation plot is 0.99833, and the performance on test data is 0.99297. The overall training performance of the dataset is 0.99639, reflecting high accuracy as indicated by its proximity to 1



Performance plot (a)



Training State Plot (b)



Regression Plot (c)

4.6 Generation of Threshold table for Training data for KDD dataset

This section focuses on evaluating the accuracy and Mean Squared Error (MSE) of the exported simulated network outputs from the trained data. The goal is to assess the network's effectiveness in detecting botnet attacks within the network. To accomplish this, four thresholds were generated using MATLAB scripts. The scripts calculated and compared these thresholds to identify the one that achieved the highest accuracy value and MSE (Mean Square error). Suitable thresholds for training datasets were obtained for training datasets Median predicted values were tested on the network outputs, resulting in the selection of four thresholds. Among these, the threshold with the lowest Mean Squared Error (MSE) was chosen as the best option. This is quite similar to choosing thresholds from the ROC (Receiver Operating Characteristics) curve which shows the relationship between the true positive rate (TPR) for the model and the false positive rate (FPR). This process involves identifying where the true positive rate (TPR) intersects with (1 -{false positive rate (FPR)}). This intersection point maximizes true positives while minimizing false negatives [50]. Among the four thresholds, the threshold of 0.95542 was selected for its high accuracy and low Mean Square Error (MSE). Although two other thresholds, 0.97184 and 0.97725, had the same MSE as the chosen threshold, they exhibited lower accuracy values. The chosen threshold was preferred because its lower MSE indicated better accuracy. Accuracy is defined as the percentage of correctly classified samples divided by the total number of samples [51]. The selected threshold was used to calculate the network outputs, with results obtained using both Microsoft Excel and MATLAB. Table 4 presents the generated threshold values along with their corresponding accuracy and MSE values.

Table 4 Threshold table generated for the training data of the KDD dataset

THE GENERATED THRESHOLDS (μ)	ACCURACY VALUE	MEAN SQUARE ERROR (MSE)
0.98005	98.30238	0.655238
0.95592	99.847619	0.6547619
0.97184	98.766667	0.6547619
0.97725	98.4190476	0.6547619

4.7 The Regression Plots Results of the KDD trained test data

As expressed in Figure 8, applying the selected threshold (0.95592) to the network outputs of the trained test data, the regression plots for training, test, and validation all show values converging at 1 after 19 iterations. This indicates that the output results are highly accurate, with all plots reaching a value of 1. Specifically, the performance of the trained dataset, the validation plot for the overall dataset, and the performance result for the testing data all show a value of 1. The convergence of all values at 1 demonstrates an exceptionally high accuracy rate for the model.

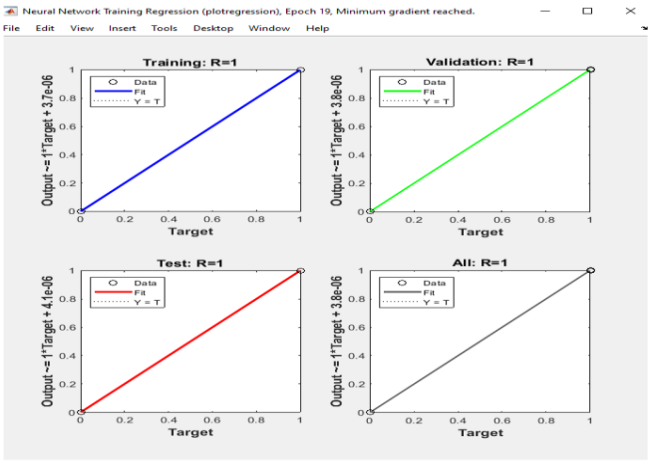


Figure 8: regression plots of the KDD-trained test data

4.8 further discussion of the KDD Trained Test Data Results

The optimal threshold (μ), determined during the training phase, was applied to the network outputs obtained from the test data. The test data was trained fifteen times, and accuracy values (%) were calculated and recorded. This process involved using the selected threshold to evaluate the network outputs by applying the IF function in Microsoft Excel. Specifically, each network output from MATLAB was copied into Excel, where the formula `[IF (A1 > 0.95592, A1, 0)]` was used. In this formula, 0.95592 is the chosen threshold from the trained data, as shown in Table 4.

Values below the threshold were set to 0, while values above the threshold were retained. After applying the threshold, the results were imported back into MATLAB, where scripts were created to calculate the average of the classified samples, thus generating the accuracy percentages. Figure 9 illustrates the application of the IF function in Excel.

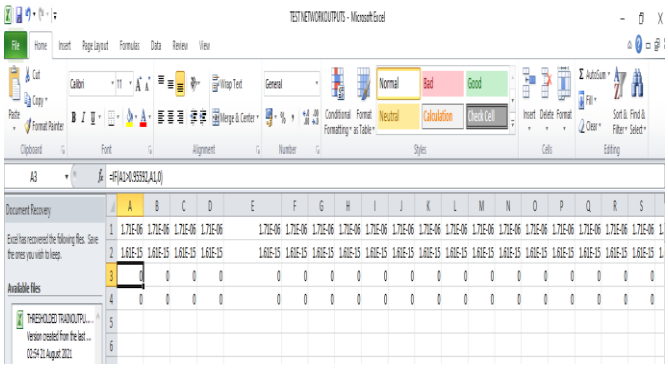


Figure 9: Microsoft Excel calculation

Table 5 presents the accuracy results of the trained test data, showing the overall average accuracy which reflects the botnet detection rate. The test dataset underwent 15 training sessions, each involving the creation of a new network. After each training session, the network outputs were exported to the workspace where the selected threshold was applied to compute the accuracy value.

Table 5: The Accuracy Values of the Trained TestData (KDD CUP'99) with the chosen Threshold

The selected threshold (μ) (0.95592)	Number of training sessions	Computed Results using the selected threshold	Accuracy value (%)
	1	(1794.9/1800) X 100	99.71667
	2	(1800/1800) X 100	100
	3	(1794.9/1800) X 100	99.71667
	4	(1800/1800) X 100	100
	5	(1794.9/1800) X 100	99.71667
	6	(1800/1800) X 100	99.94444
	7	(1800/1800) X 100	100
	8	(1800/1800) X 100	100
	9	(1800/1800) X 100	100
	10	(1800/1800) X 100	100
	11	(1794.9/1800) X 100	99.71667
	12	(1794.9/1800) X 100	99.71667
	13	(1799.0/1800) X 100	99.99444
	14	(1794.9/1800) X 100	99.71667
	15	(1798.9/1800) X 100	99.93889

The overall average represents the accuracy detection rate =

$$\frac{99.71667 + 100 + 99.71667 + 100 + 99.71667 + 99.94444 + 100 + 100 + 100 + 100 + 99.71667 + 99.71667 + 99.99444 + 99.71667 + 99.93889}{15}$$

Accuracy rate (%) = 99.87852%

Using the ANN model, an accuracy detection rate of 99.87852% was achieved, demonstrating a high effectiveness in detecting Botnet attacks. To further validate these results, several parameters were considered:

a) **Correct Rate** (0.99877): This rate is calculated as the ratio of correctly classified samples to the total classified samples. It indicates a high accuracy in detecting Botnet

attacks, with a value close to 1, affirming the model's effectiveness.

{Correct Rate} = $\frac{1797.8}{1800} = 0.99877$

b) **Error Rate (1.2222×10^{-3}):** This is derived from dividing the number of incorrectly classified samples by the total classified samples. A lower error rate signifies fewer errors in detection, indicating high model performance.

{Error Rate}= $\frac{2.2}{1800} = 1.2222 \times 10^{-3}$

c) **Last Correct Rate** (0.99877): This reflects the last recorded performance of the classifier, calculated as the ratio of correctly classified samples to the total classified samples.

d) **Last Error Rate** (1.2222×10^{-3}): This represents the last recorded error rate, calculated as the ratio of incorrectly classified samples to the total classified samples.

e) **Inconclusive Rate** (0): This value is obtained by dividing the number of nonclassified samples by the total number of samples. A rate of 0 indicates no unclassified samples, proving the model's ability to classify all data.

f) **Classified Rate** (1): Calculated as the ratio of classified samples to the total number of samples. A value of 1 indicates that all samples were classified accurately. {Classified Rate} = $\frac{1800}{1800} = 1$

g) **Sensitivity** (0.9999992): Sensitivity measures how well the model identifies true positives. It is calculated as the ratio of correctly classified positive samples to the sum of true positive and false negative samples.

{Sensitivity} = $\frac{898.9}{898.9+0.00073} = 0.9999992$

h) **Specificity** (0.99999993): Specificity assesses how well the model identifies true negatives. It is calculated as the ratio of correctly classified negative samples to the sum of true negative and false positive samples. {Specificity}= $\frac{898.9}{898.9+0.00049} = 0.9999995$

i) **Positive Predictive Value** (0.9999995): This is calculated as the ratio of correctly classified positive samples to the total positive classified samples.

j) **Negative Predictive Value** (0.9999992): This is obtained by calculating the ratio of correctly classified negative samples to the total negative classified samples.

To further validate the effectiveness of the accuracy rate in detecting botnet attacks using the Artificial Neural Network (ANN) model, a Confusion Matrix was utilized. The Confusion Matrix is a tool designed to evaluate the performance of a classification model by providing a detailed overview of its accuracy and error rates [53]. It allows for a comprehensive interpretation of how well the classification model performs by comparing the predicted values with the actual values.

The Confusion Matrix for the test data is shown in Figure 10, illustrating how the true class labels are plotted against the predicted class labels obtained from the trained model. The matrix results are displayed diagonally, indicating a 100% accuracy rate with no misclassification of data. This results in a True Positive Rate (TPR) of 100% and zero false negatives. Specifically, the Confusion Matrix confirms the accurate classification of:

a. Predicted class “A” (Back attack)
b. Predicted class “B” (Neptune attack)
c. Predicted class “C” (Smurf attack)
d. Predicted class “D” (Normal traffic)

Each class was correctly classified with no errors, demonstrating the high effectiveness of the neural network model in detecting and classifying botnet attacks

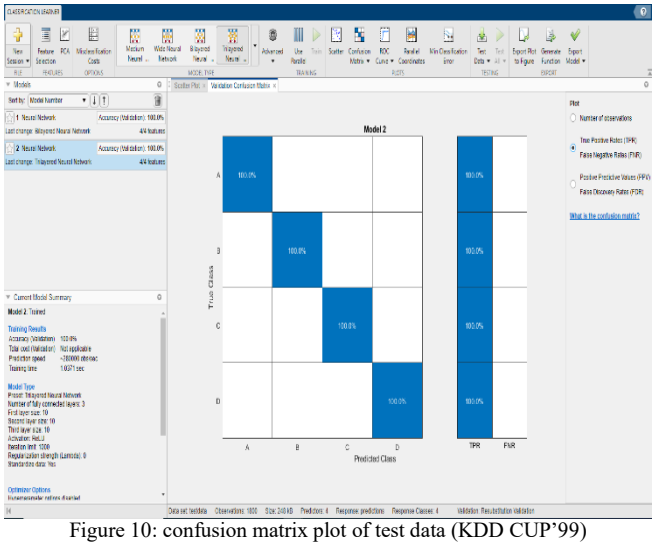


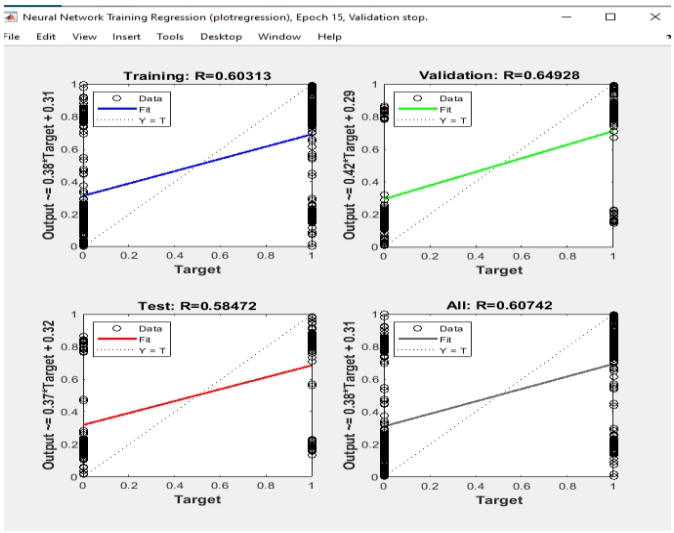
Figure 10: confusion matrix plot of test data (KDD CUP’99)

4.9 The graphical plots results of the UNSW_NB15 trained test data

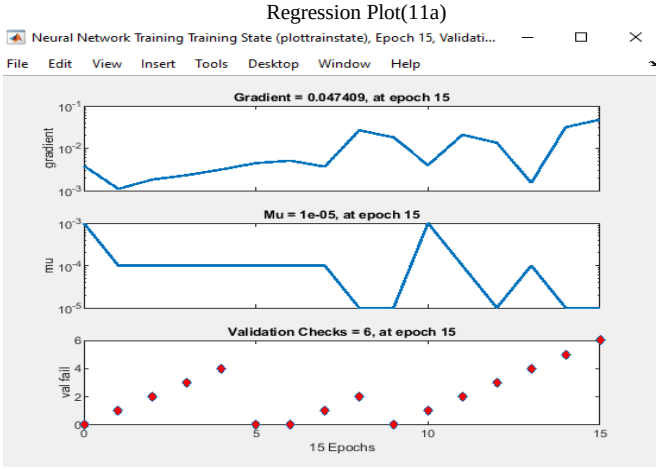
The same procedures used for processing the KDD CUP’99 dataset were applied to the UNSW_NB15 dataset. This included importing the data into MATLAB, transposing, creating a network using nntool, training, and simulating to obtain network outputs. The network outputs were analyzed using performance plots, training state plots, and regression plots.

Figure 11 illustrates the graphical plots obtained from training the data with the ANN model:

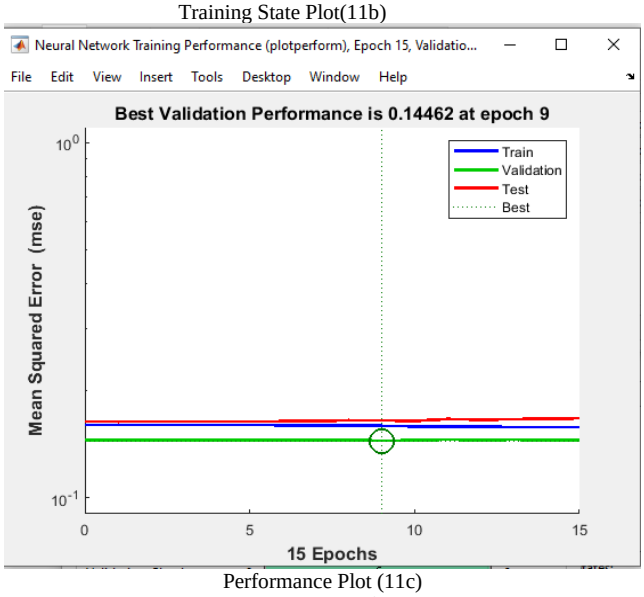
1) **Figure 11a**: The regression plot at epoch 15 shows a performance of 0.60213 for the trained dataset, 0.64928 for the validation plot, 0.58472 for the testing data, and an overall result of 0.60742 for general training. These values indicate that the performance of the ANN model on the UNSW_NB15 dataset is moderate but not as high as the results from the KDD CUP’99 dataset.
2) **Figure 11b**: The training state plot reveals a validation check of 6, a gradient convergence at 0.047409, and a control parameter (μ) of 1e-05 at epoch 15, which is close to zero.
3) **Figure 11c**: The performance plot indicates that validation of the trained data stopped at iteration 9 and remained steady with a very low mean square error (MSE), close to zero, reflecting the model’s effectiveness on this dataset, though not as high as in the KDD CUP’99 dataset.



Regression Plot(11a)



Training State Plot(11b)



Performance Plot (11c)

Figure 11: graphical plots interpretation of the Train data (UNSW-NB15 dataset)

To further assess the performance of the model on the UNSW_NB15 dataset, the accuracy of the trained data was evaluated using the Classification Learner app in MATLAB, following the same process applied to the KDD dataset. The confusion matrix provides a detailed understanding of the performance results. As shown in Figure 11, the overall accuracy of the model on the training data was 92.2%. Specifically, 92.2% of DoS attacks were correctly classified, while 7.8% were misclassified (false negative rate). For normal traffic, 92.5% was accurately classified, with 6.5% incorrectly classified (false negative rate). These results demonstrate the effectiveness of the model on the UNSW_NB15 dataset.

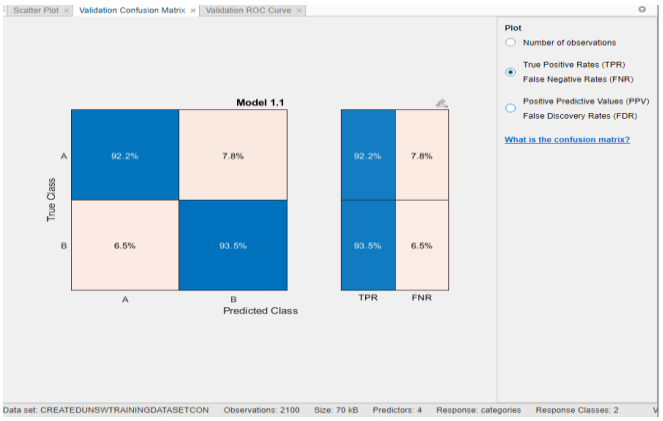


Figure 12: confusion matrix plot results for train-data (UNSW-NB15)

Table 6 presents a comparison of the results obtained from the designed ANN model with those from various studies using different algorithms and datasets. The designed ANN model achieved an accuracy of 99.87852% with the KDD CUP'99 dataset and 96% with the UNSW_NB15 dataset. These results are compared with those from Qazi [54], who used CNN (Convolutional Neural Networks) and RNN (Recurrent Neural Network) algorithms on the CICIDS-2018 dataset, achieving an average accuracy of 98.90%, an F-measure of 99.03, a precision of 98.64%, and a recall of 99.15%.

Additionally, the ANN model's accuracy results for the KDD dataset (99.87852%) and UNSW_NB15 dataset (96%) are compared with those obtained by Bhavsar[55], who developed the Pearson-Correlation Coefficient - Convolutional Neural Networks (PCC-CNN) algorithm for evaluating three datasets: NSL-KDD, CICIDS-2017, and IOTID20, achieving a detection accuracy of 99.89% and a low misclassification rate of 0.001.

The proposed ANN model is also compared with the results from Ayantayo [56], who employed Early-fusion, late-fusion, and late-ensemble learning models using feature fusion with fully connected deep networks to analyze the UNSW-NB15 and NSL-KDD datasets. This approach yielded an accuracy of 86.81%, a recall of 86.80%, and a precision of 86.86%. In contrast, the proposed model achieved 99.87852% accuracy with the KDD CUP'99 dataset and 96% accuracy with the UNSW_NB15 dataset.

Finally, a comparison is made with the results from Srinivasan [57], who used an Extreme Learning Machine (ELM), Support Vector Machine (SVM), and Convolutional Neural Network (CNN) with the Ensemble Classifier Algorithm with Stacking Process (ECASP) to analyze the Cyber Clean Center (CCC) dataset. This method achieved

94.08% accuracy, 86.5% sensitivity, 85.68% specificity, and a 78.24% F-measure. The proposed ANN model outperformed these results, highlighting its effectiveness. This shows that the proposed model performed better in terms of its accuracy results on the two datasets applied.

Table 6. Comparison of performance results of the existing related works with the proposed model

Author(s)	Applied Methods (algorithms) classifier	Name of Dataset (s) employed	Performance (Accuracy) Results
[54]	CNN and RNN	CICIDS-2018	The model achieved an average accuracy of 98.90%, an F-measure of 99.03, a precision of 98.64%, and a recall of 99.15%.
[55]	Pearson-Correlation Coefficient - Convolutional Neural Networks (PCC-CNN)	NSL-KDD, CICIDS-2017, and IOTID20	Similar accuracies of 98%, 99%, and 98% from the KNN and CART models across three datasets were achieved. However, the proposed PCC-CNN model demonstrated superior performance with a detection accuracy of 99.89% and a low misclassification rate of 0.001.
[56]	Early-fusion, late-fusion, and late-ensemble learning models using feature fusion with fully connected deep networks	UNSW-NB15 and NSL-KDD	The model achieved an accuracy of 86.81%, recall (86.80%), and 86.86% precision.
[57]	Extreme Learning Machine (ELM), Support Vector Machine (SVM), and Convolutional Neural Network (CNN) with the proposed Ensemble Classifier Algorithm with Stacking Process (ECASP).	Cyber Clean Center (CCC) dataset	the method achieves 94.08% accuracy, 86.5% sensitivity, 85.68% specificity, and 78.24% F-measure
This paper	Artificial Neural Network (ANN) using Levenberg-Marquardt Back propagation algorithm	KDD Cup'99 datasets & UNSW_NB15 dataset	99.87852% was achieved using the selected threshold and 100% was achieved using the confusion matrix plot for the KDD dataset while a 96% accuracy value was obtained using the UNSW_NB15 dataset.

5. Conclusion and Future Scope

The successful implementation of the Artificial Neural Network (ANN) model, which involved meticulous adherence to all procedural steps, demonstrates its effectiveness in detecting and classifying botnet attacks from normal traffic based on accuracy results. This effectiveness is attributed to the meticulous collection and extraction of features from the KDD CUP'99 and UNSW_NB15 datasets using Python (PyCharm software). The model was trained on 70% of the datasets and tested in MATLAB 2021a, with the results validated using a confusion matrix. The ANN model achieved an accuracy of 99.87852% using the selected threshold and 100% using the confusion matrix for the KDD dataset, with a very low false rate close to zero. For the UNSW_NB15 dataset, the model achieved a 96% accuracy rate. The study focused on DoS attack types in both datasets, specifically selecting Back, Neptune, and Smurf attacks from the KDD dataset and only Dos attacks in the UNSW_NB15 dataset which produced high-performance results.

Since the level of cyber-attacks and mode of operation by the malicious hackers change from time to time (such as zero-day attacks), which sometimes cannot be easily identified by already existing detection techniques and models, therefore, future research should explore more dynamic and accurate network models or frameworks for real-time identification of various cyber-attacks, thus, lessening their impacts on organizational and private networks.

Another aspect that future research work should focus on is evaluating different datasets or attack class types from the KDD CUP'99 and UNSW_NB15 datasets to determine their performances using the proposed model as well as testing them on other developed algorithms.

Data Availability
All details about the processing of the training and testing data for the two datasets for the proposed network model are available on request

Conflict of Interest
The author asserts that no party has exerted undue influence on the publication of this work.

Funding Source
None

Authors' Contributions
The author independently completed the entire work..

Acknowledgments
I am deeply grateful to God Almighty for His divine guidance, which has illuminated my path and inspired my work. Furthermore, I would like to express my gratitude to the International Journal of Scientific Research in Computer Science and Engineering for their invaluable feedback and constructive suggestions, which have significantly enhanced the clarity and visibility of my research efforts.

References

[1] Forti and S. Héroux, "Limited usefulness of firm-provided cybersecurity information in institutional investors' investment analysis," *Information and Computer Security*, vol. 31, Issue 3, pp. 108-123, 2023.

[2] M. R. Kadri, A. Abdelli, J. B. Othman and L. Mokdad, "Survey and classification of Dos and DDos attack detection and validation approaches for IoT environments," *Internet of Things*, vol. 25, Issue101021, pp. 1-44, 2024.

[3] H. P. S and J. R, "A Survey on the Applications of Machine Learning in Identifying Predominant Network Attacks," *International Journal of Scientific Research in Computer Science and Engineering*, vol. 11, Issue 5, pp. 16-22, 2023.

[4] G. Onuh and P. Owa, "Implementation of Slowloris Distributed Denial of Service (DDOS) Attack on Web Servers," *International Journal of Scientific Research in Computer Science and Engineering*, vol. 10, Issue 2, pp. 11-15, 2022.

[5] S. Srinivasan and P. Deepalakshmi, "ENetRM: ElasticNet Regression Model based malicious cyber-attacks prediction in real-time server," *Measurement: Sensors*, vol. 25, Issue 100654, pp. 1-10, 2023.

[6] P. Bisht and M. S. Rauthan, "Machine Learning and Natural Language Processing Based Web Application Firewall for Mitigating Cyber Attacks in Cloud," *International Journal of Scientific Research in Computer Science and Engineering*, vol. 11, Issue 3, pp. 01-15, 2023.

[7] L. D'hooge, M. V. Wauters, F. D. Turck and B. Volckaert, "Investigating Generalized Performance of Data-Constrained Supervised Machine Learning Models on Novel, Related Samples in Intrusion Detection," *Sensors*, vol. 23, Issue 4, pp. 1-39, 2023.

[8] D. Pecioski, V. Gavriloski, S. Domazetovska, and A. Ignjatovska, "An overview of reinforcement learning techniques," in *2023 12th Mediterranean Conference on Embedded Computing (MECO)*, Budva, Montenegro, 2023.

[9] K. A. Okewale, I. R. Idowu, B. S. Alobalorun and F. A. Alabi, "Effective Machine Learning Classifiers for Intrusion Detection in Computer Network," *International Journal of Scientific Research in Computer Science and Engineering*, vol. 11, Issue 2, pp. 14-22, 2023.

[10] D. Gibert, J. Planes, C. Mateu and Q. Le, "Fusing feature engineering and deep learning: A case study for malware," *Expert Systems With Applications*, vol. 207, Issue 117957, pp. 1-18, 2022.

[11] S. Kayikci and T. M. Khoshgoftaar, "Blockchain meets machine learning: a survey," *Journal of Big Data*, vol. 11, Issue 9, pp. 1-29, 2024.

[12] M. H. Thwaini, "Anomaly Detection in Network Traffic using Machine Learning for Early Threat," *Data & Metadata*, vol. 1, Issue 34, pp. 1-16, 2022.

[13] S. Sheng and X. Wang, "Network traffic anomaly detection method based on chaotic neural network," *Alexandria Engineering Journal*, vol. 77, pp. 567-579, 2023.

[14] H. Liu and H. Wang, "Real-Time Anomaly Detection of Network Traffic Based on CNN," *Symmetry*, vol. 15, Issue 6, pp. 1-21, 2023.

[15] F. Hu, S. Zhang, X. Lin, L. Wu, N. Liao and Y. Song, "Network traffic classification model based on attention mechanism and spatiotemporal features," *EURASIP Journal on Information Security*, vol. 1, Issue 6, pp. 1-25, 2023

[16] I. H. Ji, J. H. Lee, M. J. Kang, W. J. Park, S. H. Jeon and J. T. Seo, "Artificial Intelligence-Based Anomaly Detection Technology over Encrypted Traffic: A Systematic Literature Review," *Sensors*, vol. 24, Issue 3, pp. 1-30, 2024.

[17] W. Hu, L. Cao, Q. Ruan, and Q. Wu, "Research on Anomaly Network Detection Based on Self-Attention Mechanism," *Sensors*, vol. 23, Issue 11, pp. 1-17, 2023.

[18] K. Lu, "Network Anomaly Traffic Analysis," *Academic Journal of Science and Technology*, vol. 10, Issue 3, pp. 65-68, 2024.

[19] Z. Dang, Y. Zheng, X. Lin, C. P. Q. Chen and X. Gao, "Semi-Supervised Learning for Anomaly Traffic Detection via Bidirectional Normalizing Flows," *arXiv*, vol. 1, pp. 1-14, 2024.

[20] S. Zehra, U. Faseeha, H. J. Syed, F. Samad, A. O. Ibrahim, A. W. Abulfaraj and W. Nagmeldin, "Machine Learning-Based Anomaly Detection in NFV: A Comprehensive Survey," *Sensors*, vol. 23, Issue 11, pp. 1-26, 2023.

[21] S. Padhiar and R. Patel, "Performance evaluation of botnet detection using machine learning techniques," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, Issue 6, pp. 6827-6835, 2023.

[22] M. Al-farttoosi and H. Abdulkader, "Botnet Mobile Detection Using Machine & Deep Learning Techniques," in *2022 Iraqi International Conference on Communication and Information Technologies (IICCIT)*, Basrah, Iraq, 2022.

[23] M. Swami, A. Yadnik, A. Jagtap, K. Bhilare and M. Wagh, "BOTNET DETECTION USING VARIOUS MACHINE LEARNING ALGORITHMS: A REVIEW," *International Research Journal of Engineering and Technology (IRJET)*, vol. 09, Issue 12, pp. 125-132, 2022.

[24] C. Joshi, V. Bharti and R. K. Ranjan, "Botnet Detection Using Machine Learning Algorithms," in *Proceedings of the International Conference on Paradigms of Computing, Communication and Data Sciences*, Singapore, 2021.

[25] J. Forough, "Anomaly Detection and Resolution for Edge Clouds," in *Machine Learning for Anomaly Detection*, Sweden, IEEE, 2024, pp. 11-25.

[26] S. A. Hussein and S. R. Répás, "Enhancing Network Security through Machine Learning-Based Anomaly Detection Systems," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, Issue 215, pp. 1929-1935, 2024.

[27] M. Landauer, S. Onder, F. Skopik and M. Wurzenberger, "Deep learning for anomaly detection in log data: A survey," *Machine Learning with Applications*, vol. 12, Issue 100470, pp. 1-19, 2023.

[28] M. ALI, M. S. M. F. MUSHTAQ, SULTAN, M. S., and I. ASHRAF, "Hybrid Machine Learning Model for Efficient Botnet Attack Detection in IoT Environment," *IEEE ACCESS*, vol. 1, Issue 1, pp. 1-19, 2024.

[29] J. Ashraf, M. Keshk, N. Moustafa, M. Abdel-Basset, H. Khurshid, A. D. Bakhshi and R. R. Mostafa, "IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities," *Sustainable Cities and Society*, vol. 72, Issue 103041, 2021.

[30] C. G. Cordero, E. Vasilomanolakis, A. Wainakh, M. Mühlhäuser, and S. N. Tehrani, "On Generating Network Traffic Datasets with Synthetic Attacks for Intrusion Detection," *ACM Transactions on Privacy and Security*, vol. 24, Issue 2, pp. 1-39, 2021.

[31] E. P. VALENTINI, G. P. R. FILHO, R. E. D. GRANDE, C. M. RANIERI, L. A. P. JÚNIOR and R. I. MENEGUETTE, "A Novel Mechanism for Misbehavior Detection in Vehicular Networks," *IEEE*, vol. 11, pp. 68113-68126, 2023.

[32] A. Z. Umar and Y. Galadima, "Detecting Anomalies In Network Traffic Using a Hybrid of Linear-based and Tree-based Feature Selection Approaches," in *International Conference on Computing and Advances in Information Technology (ICCAIT 2023)*, Ahmadu Bello University, Zaria, Nigeria, 2023.

[33] S. Choudharya and N. Kesswani, "Analysis of KDD-Cup'99, NSL-KDD, and UNSW-NB15 Datasets using Deep Learning in IoT," in *International Conference on Computational Intelligence and Data Science (ICCIDS 2019)*, Rajasthan, India, 2020.

[34] S. More, M. Idrissi, H. Mahmoud and A. T. Asyhar, "Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis," *aligorithms*, vol. 17, Issue 64, pp. 1-16, 2024.

[35] T. A. S. Srinvias, A. D. Donald, M. Sameena, K. Rekha and I. D. Srihith, "Unlocking the Power of Matlab: A Comprehensive Survey," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 3, Issue 1, pp. 20-31, 2023.

[36] Y. SAHLI, "A comparison of the NSL-KDD dataset and its predecessor the KDD Cup'99 dataset," *International Journal of Scientific Research and Management (IJSRM)*, vol. 10, Issue 4, pp. 832-839, 2022.

[37] I. Dutt, S. Borah and I. K. Maitra, "Pre-Processing of KDD'99 & UNSW-NB Network Intrusion Datasets," *Turkish Journal of*

Computer and Mathematics Education, vol. 12, Issue 11, pp. 1762-1776, 2021.
[38] M. Gelgi, Y. Guan, S. Arunachala, M. S. S. Rao, and N. Dragoni, "Systematic Literature Review of IoT Botnet DDOS Attacks and Evaluation of Detection Techniques," *Sensors,* vol. 24, Issue 11, pp. 1-37, 2024.
[39] O. Valenzuela, A. Catala, D. Anguita and I. Rojas, "New Advances in Artificial Neural Networks and Machine Learning Techniques," *Neural Process Letters,* vol. 55, Issue 1, pp. 5269-5272, 2023.
[40] S. Hartanto, "The Impact of Smurf Attack on Web Server in Communication Network and its Preventions," *International Journal of Sustainable Applied Sciences (IJSAS),* vol. 1, Issue 1, pp. 35-46, 2023.
[41] M. B. Anley, A. Genovese, D. Agostinello and V. Piuri, "Robust DDoS attack detection with adaptive transfer learning," *Computers & Security,* vol. 144, Issue 103962, pp. 1-10, 2024.
[42] M. M. Abualhaj, A. A. Abu-Shareha, M. O. Hiari, Y. Alrabanah, M. Al-Zyoud and M. A. Alsharaiah, "A Paradigm for DoS Attack Disclosure using Machine Learning Techniques," *(IJACSA) International Journal of Advanced Computer Science and Applications,* vol. 13, Issue 3, pp. 192-200, 2022.
[43] J. H. Yousif and H. A. Kazem, "Prediction and evaluation of photovoltaic-thermal energy systems production using artificial neural network and experimental dataset," *Case Studies in Thermal Engineering,* vol. 27, Issue 101297, pp. 1-13, 2021.
[44] W. Ahmed, A. Chaudhary, and G. Naqvi, "Role of Artificial Neural Networks in AI," *Neuro Quantology,* vol. 20,. Issue 13, pp. 3365-3373, 2022.
[45] Y. D. Jian Liu, Y. Liu, L. Chen, Z. Hu, P. Wei and Z. Li, "A logistic-tent chaotic mapping Levenberg Marquardt algorithm for improving positioning accuracy of grinding robot," *Scientific Reports,* vol. 14, Issue 9649, pp. 1-15, 2024.
[46] M. K. Hasan, A. A. Habib, S. Islam, N. Safie and B. Pandey, "DDoS: Distributed denial of service attack in communication standard vulnerabilities in smart grid applications and cyber security with recent developments," *Energy Reports,* vol. 9, Issue 10, pp. 1318-1326, 2023.
[47] C. G. Udomboso and O. O. Ilori, "A DERIVED HETEROGENEOUS TRANSFER FUNCTION FROM CONVOLUTION OF SYMMETRIC HARDLIMIT AND HYPERBOLIC TANGENT SIGMOID TRANSFER FUNCTIONS," *Journal of Science and Technology,* vol. 40, Issue 1, pp. 27-37, 2022.
[48] T. Y. Li, H. Xiang, Y. Yang, J. Wang and G. Yildiz, "Prediction of char production from slow pyrolysis of lignocellulosic biomass using multiple nonlinear regression and artificial neural network," *Journal of Analytical and Applied Pyrolysis,* vol. 159, no. Issue 105286, 2021.
[49] I. Dubdub, "Pyrolysis Study of Mixed Polymers for Non-Isothermal TGA: Artificial Neural Networks Application," *Polymers,* vol. 14, Issue 2638, pp. 1-10, 2022.
[50] A. A. R. A.-c. Omar, B. Soudan and A. Altaweel, "A comprehensive survey on detection of sinkhole attack in routing over low power and Lossy network for internet of things," *Internet of things,* vol. 22, Issue 100750, 2023.
[51] E.-M. Nikolados, A. Wongprommoon, O. M. Aodha, G. Cambray and D. A. Oyarzún, "Accuracy and data efficiency in deep learning models of protein expression," *Nature communications,* vol. 13, Issue 7755, pp. 1-12, 2023.
[52] T. F. Monaghan, S. N. Rahman, C. W. Agudelo, A. J. Wein, J. M. Lazar, K. Everaert and R. R. Dmochowski, "Foundational Statistical Principles in Medical Research: Sensitivity, Specificity, Positive Predictive Value, and Negative Predictive Value," *Medicina,* vol. 57, Issue 503, pp. 1-7, 2021.
[53] L.-E. Pommé, R. Bourqui, R. Giot and D. Auber, "Relative Confusion Matrix: Efficient Comparison of Decision Models," in *2022 26th International Conference Information Visualisation (IV),* Vienna, Austria, 2022.
[54] E. U. H. Qazi, M. H. Faheem and T. Zia, "HDLNIDS: Hybrid Deep-Learning-Based Network Intrusion Detection System," *Applied Sciences,* vol. 13, Issue 4921, pp. 1-16, 2023.
[55] M. Bhavsar, K. Roy, J. Kelly and O. Olusola, "Anomaly-based intrusion detection system for IoT application," *Discover Internet of Things,* vol. 3, Issue 5, pp. 1-23, 2023.
[56] A. Ayantayo, A. Kaur, A. Kour, X. Schmoor, F. Shah, I. Vickers, P. K. and M. M. Abdelsamea, "Network intrusion detection using feature fusion with deep learning," *Journal of Big Data,* vol. 10, Issue 67, pp. 1-24, 2023.
[57] S. Srinivasan and D. P, "Enhancing the security in cyber-world by detecting the botnets using ensemble classification based machine learning," *Measurement: Sensors,* vol. 25, Issue 100624, pp. 1-7, 2023.

AUTHORS PROFILE

Grace Bunmi Akintola earned a B.Tech in Computer Science with a specialization in Cyber Security from the Federal University of Technology Minna, Niger State, Nigeria, in 2016. She then pursued an MSc in Computer Forensics and Cybersecurity from the University of Greenwich, London, United Kingdom, graduating in 2021. These educational experiences have provided her with a comprehensive understanding of cybersecurity principles, best practices, and forensic techniques. Grace is a member of the Nigeria Computer Society (NCS), the umbrella organization for all Information Technology Professionals, Interest Groups, and Stakeholders in Nigeria. Currently, she serves as an Assistant Lecturer in the Department of Cyber Security at the Nigerian Defence Academy (NDA) in Kaduna, Nigeria. In this role, she is actively involved in educating and impacting future cybersecurity professionals, fostering a cybersecurity awareness culture, and conducting field research. Her commitment to academic excellence is reflected in her continuous pursuit of knowledge and dedication to her students. Grace is interested in research, scholarly writing, and various fields in cybersecurity, such as network security, forensics, AI security, penetration testing, and more.

