Scientific Research Publishing

Journal of Computer and Communications, 2024, 12, 38-58
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Machine Learning Models for Heterogenous Network Security Anomaly Detection

Mercy Diligence Ogah[1], Joe Essien[1], Martin Ogharandukun[2], Monday Abdullahi[3]

[1]Department of Computer Science, Veritas University, Abuja, Nigeria
[2]Department of Pure and Applied Physics, Veritas University, Abuja, Nigeria
[3]Department of Computer Science, Air Force Institute of Technology, Kaduna, Nigeria
Email: mercydiligence@gmail.com, essienj@veritas.edu.ng, OgharandukunM@veritas.edu.ng, m.abdullahi@afit.edu.ng

How to cite this paper: Ogah, M.D., Essien, J., Ogharandukun, M. and Abdullahi, M. (2024) Machine Learning Models for Heterogenous Network Security Anomaly Detection. *Journal of Computer and Communications*, **12**, 38-58.
https://doi.org/10.4236/jcc.2024.126004

Received: May 23, 2024
Accepted: June 22, 2024
Published: June 25, 2024

Abstract

The increasing amount and intricacy of network traffic in the modern digital era have worsened the difficulty of identifying abnormal behaviours that may indicate potential security breaches or operational interruptions. Conventional detection approaches face challenges in keeping up with the ever-changing strategies of cyber-attacks, resulting in heightened susceptibility and significant harm to network infrastructures. In order to tackle this urgent issue, this project focused on developing an effective anomaly detection system that utilizes Machine Learning technology. The suggested model utilizes contemporary machine learning algorithms and frameworks to autonomously detect deviations from typical network behaviour. It promptly identifies anomalous activities that may indicate security breaches or performance difficulties. The solution entails a multi-faceted approach encompassing data collection, preprocessing, feature engineering, model training, and evaluation. By utilizing machine learning methods, the model is trained on a wide range of datasets that include both regular and abnormal network traffic patterns. This training ensures that the model can adapt to numerous scenarios. The main priority is to ensure that the system is functional and efficient, with a particular emphasis on reducing false positives to avoid unwanted alerts. Additionally, efforts are directed on improving anomaly detection accuracy so that the model can consistently distinguish between potentially harmful and benign activity. This project aims to greatly strengthen network security by addressing emerging cyber threats and improving their resilience and reliability.

Keywords

Cyber-Security, Network Anomaly Detection, Machine Learning, Random Forest, Decision Tree, Gaussian Naive Bayes

1. Introduction

The growing dependence on interconnected computer networks has raised the susceptibility of organizations to cyber threats and security breaches [1]. In today's fast-paced digital world, the ever-increasing amount and intricacy of network traffic have posed significant challenges for network security and anomaly detection. Conventional approaches have fallen short in effectively addressing the constantly evolving cyber risks. With the increasing sophistication of cyber-attacks and the complexity of computer networks, traditional methods of detecting abnormalities in network security are facing significant challenges [2]. Signature-based intrusion detection systems struggle to keep up with evolving cyber threats, often resulting in a high number of false positive alerts [3]. The complex patterns of typical network activity can mimic those indicating abnormalities, complicating accurate detection of security breaches. Given these difficulties, there is a crucial need for more adaptable and proactive approaches, such as utilizing Machine Learning (ML) [3]. Recent studies highlight the shortcomings of signature-based Intrusion Detection System (IDS) in detecting new and advanced attacks, as they rely on predefined patterns [1] [4]. ML models can independently learn and adapt to changing threats, identifying intricate patterns in network traffic [5]. This capability improves proactive detection of anomalies, reduces false positives, and enhances overall network security measures.

In order to fully understand the benefits of ML in network security, it is crucial to conduct thorough research that emphasizes the practical application and assessment of ML models for real-time anomaly detection [3]. This study aims to address the existing gap by developing and deploying a specialized ML model for network anomaly detection, with the ultimate goal of improving network security. Many researchers contended that the limitations of conventional techniques and the ever-changing nature of cyber risks call for a fundamental change towards utilizing ML-driven methods for detecting network anomalies [5] [6]. This study seeks to make a valuable contribution to the current body of knowledge by offering practical insights into the efficacy of ML models in bolstering network security.

Categories of Network Anomalies and Detection

Network anomalies encompass various types that challenge security and performance, each necessitating tailored detection methods. A common category is traffic spikes, which involve sudden surges in data flow [6]. These can result from legitimate causes, such as a popular event driving web traffic, or malicious activities like Distributed Denial of Service (DDoS) attacks. Effective detection differentiates between benign traffic increases, which may require additional resources, and harmful spikes, which demand mitigation strategies like traffic filtering [7]. Another significant anomaly type involves unusual protocol or port activity [8]. Deviations from standard protocol usage or unexpected port activity often indicate unauthorized applications or malicious actions. For example,

communication via non-standard ports or unknown protocols can signal potential security threats. Intrusion Detection Systems (IDS) are essential for monitoring and identifying these abnormal activities, allowing for timely response to potential risks [9].

Packet loss, another critical anomaly, occurs when data packets fail to reach their destinations, often due to network congestion, hardware failures, or poor connections [7]. This can severely impact application performance and user experience. Network monitoring solutions that track packet loss metrics are vital for diagnosing the root causes and implementing corrective measures to maintain network reliability. Additionally, anomalies in user behaviour, such as abnormal login times or unauthorized data transfers, can indicate insider threats or compromised accounts [9]. It is important to closely monitor these behaviours, as they have the potential to culminate in serious security breaches if undetected. Security Information and Event Management (SIEM) are systems pivotal in correlating these anomalies across various data sources, providing comprehensive insights and enabling proactive risk management [8]. Finally, the early detection of zero-day attacks, which exploit previously unknown vulnerabilities, underscores the importance of network anomaly detection [5]. These attacks lack predefined signatures, making traditional detection methods ineffective. Anomaly detection systems that identify deviations from normal behaviour patterns provide a crucial defence layer. Overall, robust anomaly detection mechanisms enhance network security, compliance with regulatory standards, and operational resilience by identifying and addressing diverse network anomalies promptly and effectively.

2. Related Works

Conventional methods for anomaly detection have made a substantial impact on the initial phases of safeguarding network environments by utilizing rule-based systems, statistical techniques, and signature-based detection mechanisms [6]. Rule-based anomaly detection utilizes pre-established sets of rules to detect anomalies from anticipated network behaviour [9]. Mothukuri, *et al*., [10], focused on cognitive architectures and problem-solving, created the notion of rule-based systems. These methods have been extensively utilized to accurately detect particular patterns linked to recognized assaults and abnormalities [10]. Although rule-based systems are designed and implemented in a basic manner, they often have difficulties in adapting to new threats and tend to generate a significant number of false positives due to their rigid structure [11].

Statistical approaches, which are a conventional approach, consist of creating baseline models that represent normal network behaviour and identifying any deviations from these baselines [7]. Statistical techniques, such as mean, median, and standard deviation analysis, have been used to analyse network traffic and have achieved a satisfactory level of accuracy in finding outliers [10]. Nevertheless, these approaches frequently encounter difficulties in dealing with the dy-

namic and developing characteristics of network activities, which diminishes their effectiveness in countering complex and innovative attacks. Signature-based detection relies on a repository of established attack signatures [11]. Network traffic that corresponds to a predetermined signature is identified as malicious. This method is efficient in dealing with acknowledged dangers, but it faces difficulties when it comes to zero-day attacks and can be readily circumvented by polymorphic or encrypted malware. Anton *et al*. [12] work in developing intrusion detection systems, which frequently utilize signature-based detection approaches, showcases the effectiveness and constraints of these systems.

Traditionally, traditional techniques including rule-based, statistical, and signature-based anomaly detection have had a crucial role in network security [5]. However, these techniques face major difficulties in keeping up with the ever-changing cyber threats. Rule-based systems exhibit inflexibility and a tendency to produce incorrect positive results, whilst statistical methods have difficulties in handling dynamic network activity, and signature-based systems are insufficient in countering emerging threats [6]. The existence of these constraints emphasizes the necessity for the implementation of more advanced anomaly detection algorithms. Traditional approaches depend on predetermined thresholds and manually designed rules, which makes them simple but frequently inadequate for dealing with intricate, evolving data. This typically results in elevated rates of false positives and limited scalability when confronted with extensive datasets [12]. In contrast, machine learning methods provide a data-centric alternative that may acquire knowledge from various datasets using techniques such as supervised learning with neural networks and unsupervised clustering [8]. These methods are capable of identifying complex patterns in data with a high number of dimensions [3]. Nevertheless, they necessitate significant processing resources and frequently lack interpretability. This comparison underscores the compromises between conventional and contemporary methods in network security, underlining the necessity for sophisticated, flexible solutions.

3. Method

The research utilizes the machine learning process to create and evaluate an ML model that improves the identification of anomalies in network security. This methodology entails collecting pertinent datasets for the purpose of training and evaluating the ML model, choosing appropriate machine learning algorithms, and optimizing model parameters. Furthermore, it involves a thorough assessment using well-established performance indicators to verify the efficiency of the ML model in identifying network irregularities. The ML technique aids in the optimization and improvement of anomaly detection systems by methodically organizing problems and creating mathematical models [13]. This methodical procedure guarantees that the ML models are customized to successfully tackle the specific obstacles in network security. The main steps in the technique involve issue conceptualization, data collection and pre-processing, feature selec-

tion and engineering, model development using TensorFlow, model training and optimization, evaluation metrics, model validation and testing, and implementation.

When developing the ML model for the network anomaly detection system, Kaggle was relied upon, a well-known platform for sharing datasets and machine learning resources, to collect the primary source of network packet data. For this work, the dataset used was the KDD Cup 1999 dataset, available on Kaggle [14]. The KDD Cup 1999 dataset from Kaggle has gained significant recognition and is widely used in the cybersecurity and machine learning communities. This dataset is derived from the 1998 DARPA Intrusion Detection Evaluation Program and covers a wide range of network traffic scenarios, including both normal and anomalous activities [14]. The extensive use of this technology in research and industry settings establishes a strong basis for evaluating ML-based anomaly detection systems. In addition, Kaggle's platform provides extra resources like forums, kernels, and competitions, allowing researchers and practitioners to work together, exchange ideas, and improve their methods. Utilizing the KDD Cup 1999 dataset from Kaggle provides access to top-notch data, along with a helpful community and a plethora of additional resources to improve the development and evaluation of the ML model for network anomaly detection. The dataset contains a significant amount of network traffic data that was generated in a simulated environment. It captures a wide range of attacks and normal activities. During the research, a comprehensive collection of information was gathered and consolidated to inform the development process. The consolidation process involved preprocessing and cleaning data collected and verifying and removing errors and storing it in a single location.

3.1. Model Selection

The work of designing and implementing aMachine Learning (ML) model for detecting network security anomalies involves a meticulous blend of data collection, statistical analysis, modeling techniques, and machine learning algorithms [15]. The process begins with gathering diverse network data streams and subjecting them to rigorous preprocessing to ensure accuracy and reliability. This is followed by feature selection and engineering, which use statistical methods to identify and transform relevant features that capture the complexities of network behavior. These foundational steps are crucial for creating robust machine learning models capable of effectively detecting network anomalies.

Among the various machine learning algorithms employed, Gaussian Naïve Bayes, Decision Tree, Random Forest, K-Nearest Neighbors (KNN), and Neural Networks each offer distinct advantages and face unique challenges in network anomaly detection [16]. Gaussian Naïve Bayes is a probabilistic classifier that assumes feature independence and Gaussian distribution, allowing it to swiftly calculate the probability of network data being normal or anomalous based on features like packet size and protocol type [15]. In contrast, Decision Tree clas-

sifiers create a tree-like structure by recursively splitting data into subsets based on feature values, forming interpretable rules for detecting anomalies such as unusual source IPs or uncommon protocols. Random Forest, an ensemble learning technique, combines multiple decision trees to enhance predictive accuracy and can handle large data sets and high-dimensional feature spaces, making it particularly adept at identifying subtle anomalies in complex network traffic [17].

3.2. Model Design and Implementation

The process of designing and implementing the model for network anomaly detection followed an extensive method that encompassed multiple stages [18]. The challenge of identifying irregularities in network traffic was first specified. Then, data from Kaggle, including network logs and packet data, was collected with great attention to detail. This ensured that the data was accurately labeled, which is crucial for training an effective model [3]. Afterwards, the gathered data went through thorough preprocessing, which involved cleaning, normalization, and feature extraction in order to obtain pertinent features that are suitable for detecting anomalies. The crucial step in this project was the careful selection of suitable ML models, specifically focusing on supervised machine learning approaches such as deep learning models [18]. These models consist of multiple layers of interconnected neurons that have the ability to learn intricate patterns in data. In this work, the architecture used was convolutional neural networks (CNNs).

The model was trained by carefully adjusting hyperparameters and using cross-validation approaches to avoid overfitting. The model's performance was assessed using different measures such as precision, recall, and F1-score. In summary, the process of designing and implementing the model followed a methodical and repetitive approach, resulting in a dependable solution for detecting network anomalies. The following is the output of the different algorithms/classifiers used in the model to identify network security abnormalities, along with their respective performance evaluation results.

4. Results and Simulations

This section presents the results and modeling outcomes of our study on Machine Learning (ML) Models for Robust Network Security Anomaly Detection. Leveraging advanced machine learning techniques, models capable of effectively detecting anomalies in network traffic are developed, thereby enhancing overall security measures [19]. The analysis encompasses the evaluation of various ML models, including Gaussian Naive Bayes Classifier, Decision Tree Classifier and Random Forest Classifier approaches, to assess their performance in accurately identifying abnormal network behavior. Through rigorous modelling and validation, insights into the effectiveness and limitations of different ML techniques in bolstering network security and mitigating cyber threats is provided.

4.1. Data Transposition

Data transposition is the process of rearranging the structure of a dataset by interchanging its rows and columns [17]. Essentially, the dataset's rows are transformed into columns, and the columns are transformed into rows.

The transposition and preprocessed dataset shown in **Figure 1** serves several justifications within data analysis and machine learning contexts. Firstly, altering the orientation of the dataset can facilitate analysis or display, making it more convenient to comprehend specific patterns or connections [16]. For instance, arranging attributes as columns and observations as rows, or vice versa, may enhance interpretability. Additionally, certain machine learning models or algorithms may necessitate data to be in a particular format. Transposing the dataset allows for meeting such requirements, especially when the original arrangement does not align with the desired format [17].

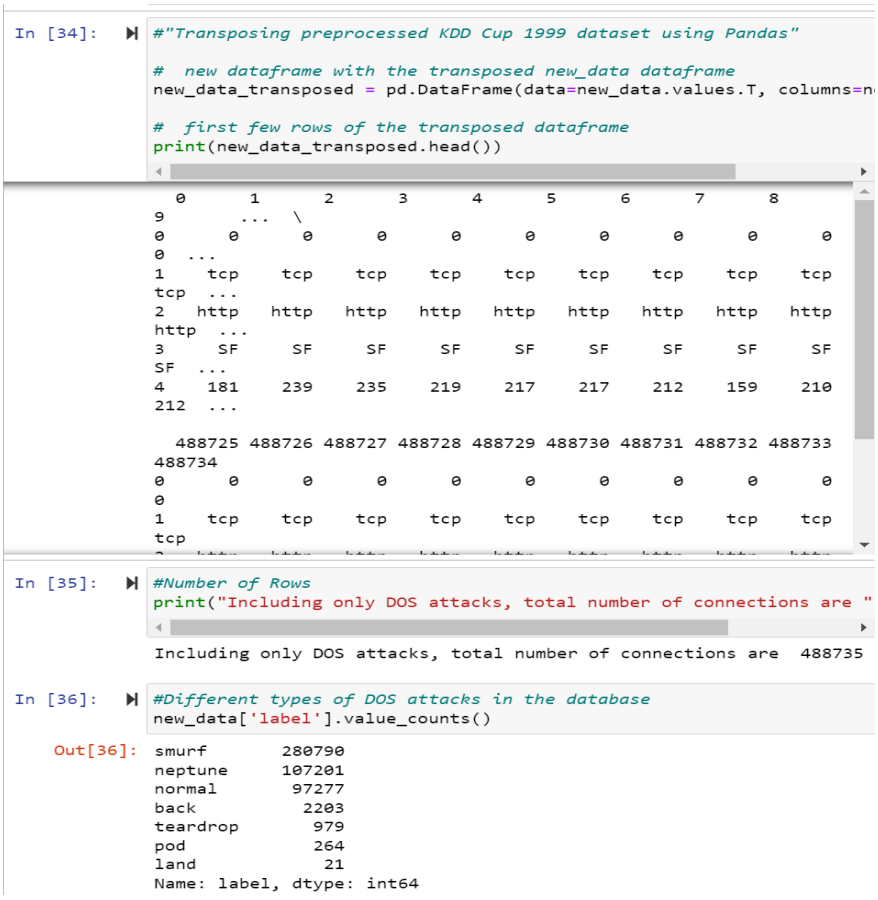

Figure 1. Transposed preprocessed dataset using Pandas.

Furthermore, transposing enables the alignment of datasets with varying orientations or structures for further analysis or modeling [18]. In the case of the KDD dataset, preprocessing precedes transposition using Pandas' DataFrame.transpose() method. This restructuring involves transforming attributes (columns) into rows and observations into columns, thereby accommodating di-

verse analysis needs. The resultant transposed DataFrame, new_data_transposed, is then analyzed to reveal its altered structure. Ultimately, transposing the dataset offers multifaceted benefits, ranging from simplifying analysis to fulfilling model prerequisites or merging data from disparate sources, contingent upon the distinct requirements of the research or application [20].

4.2. Distribution of Protocol Type

The distribution of protocol type pertains to the dissemination or allocation of various communication protocols within a dataset or network environment. In network analysis, the distribution of protocol types refers to the relative frequencies or proportions of different communication protocols, such as ICMP (Internet Control Message Protocol), TCP (Transmission Control Protocol), UDP (User Datagram Protocol), and others [18].

Figure 2 illustrates the distribution of protocol types within the KDD dataset, The bar chart offers insights into three main aspects: Protocol Type, denoting the network protocol used for packet transmission, with ICMP, TCP, and UDP being predominant; Number of Packets, indicating the absolute count of packets attributed to each protocol type, showing ICMP with the highest count followed by TCP and UDP; and Percentage of Packets, representing the proportion of packets relative to the total dataset, with ICMP constituting the largest share followed by TCP and UDP. The x-axis label, "Number-of-packets/Percentage-of-packets," contextualizes the values, incorporating both absolute counts and percentages, while the y-axis label, "Protocol type," identifies the categories, facilitating straightforward interpretation of the depicted protocol types.

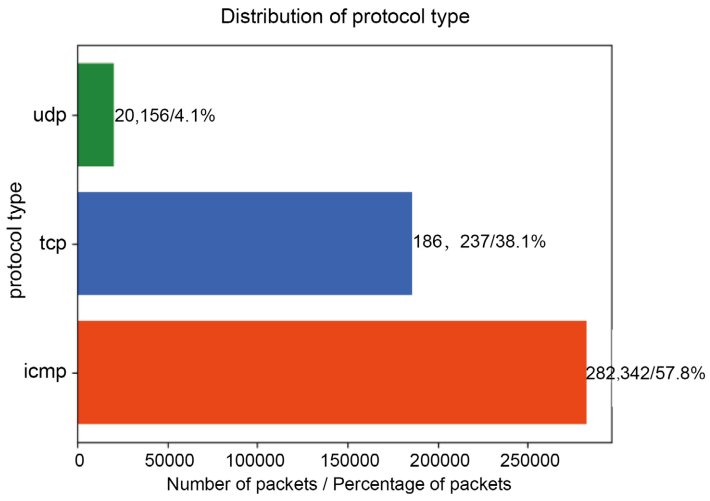

Figure 2. The distribution of protocol type.

4.3. Classification of Denial-of-Service Attacks (Dos Attacks)

Figure 3 illustrates the distribution of DoS attack types based on the number of connections observed in each attack, with the Smurf attack being the most prevalent, accounting for 57.5% of total connections. This form of cyber assault

floods the victim's network with ICMP echo request packets, targeting IP broadcast addresses to overwhelm available bandwidth and induce a denial of service. Neptune, representing 21.9% of connections, involves flooding the target's network with TCP packets to exhaust resources like bandwidth and CPU (**Figure 3**). The "normal" category, comprising 19.9% of connections, denotes benign network traffic. Other attack types include the back attack, Teardrop, Pod (or Ping of Death), and Land attack, each with varying levels of representation in the dataset.

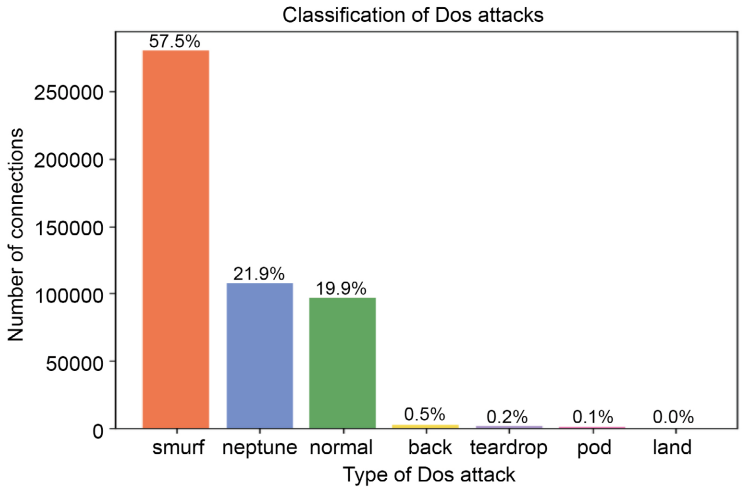

Figure 3. Presentation of the various classes of DOS attacks.

4.4. Feature Selection using Recursive Feature Elimination (RFE)

In order to improve the efficiency and efficacy of the classification model, the technique of Recursive Feature Elimination (RFE) is used in combination with a Logistic Regression model [20]. This methodology enables the selection of the most pertinent characteristics from the dataset, thereby enhancing the prediction performance of the model [21]. The Logistic Regression model is employed from the sklearn.linear_model module and set with RFE to choose the top ten features depending on their significance. Subsequently, the chosen characteristics and their corresponding rankings were obtained for subsequent examination. Further, the Extra Trees Classifier was utilized to assess the significance of features in the dataset, alongside the RFE method. This classifier offers a metric for determining the significance of each feature by evaluating its contribution to the overall prediction capability of the model [22]. The feature rankings were displayed in a descending sequence, offering a distinct comprehension of the most impactful aspects in the dataset.

4.5. Feature Selection and Data Preprocessing

In order to prepare the dataset for model training, feature selection and data preprocessing were executed subsequent to the evaluation of feature importance. Attributes deemed less significant for the classification assignment, including

"duration", "flag", and "dst_bytes", were identified and eliminated. By iteratively removing attributes from the dataset in accordance with a predefined list, a more targeted and pertinent feature set was generated. Following this, the resultant dataset was appended to a fresh file (Figure 4), to enable subsequent analysis and model training.

```
In [50]: #Writing the extracted data into new file
         new_data.to_csv("KDD_DOS_ATTACK_DATA.csv",encoding='utf-8', index=False)
         new_data.head()
```

Out[50]:

	protocol_type	service	src_bytes	count	same_srv_rate	dst_host_same_srv_rate	dst_host_same_src_port_rate	label
0	1	33	181	8	1.0	1.0	0.11	0
1	1	33	239	8	1.0	1.0	0.05	0
2	1	33	235	8	1.0	1.0	0.03	0
3	1	33	219	6	1.0	1.0	0.03	0
4	1	33	217	6	1.0	1.0	0.02	0

Figure 4. Presentation of factorized dataset.

4.6. Modelling Gaussian Naive Bayes Classifier and Performance Evaluation

1) The dataset is divided into two subsets: a training set and a testing set, using the train_test_split function from the sklearn.model_selection module. The training set is used to train the model, while the testing set evaluates its performance. The Gaussian Naive Bayes classifier algorithm is employed, chosen for its assumption of feature independence and Gaussian distribution of features, which are common in many real-world scenarios. The GaussianNB class from the sklearn.naive_bayes module is used to initialize the classifier, forming the foundation for training and evaluation. During the training phase, the classifier learns patterns and relationships within the training data by adjusting its internal parameters through iterative optimization, minimizing discrepancies between predicted and actual class labels. Performance evaluation follows, where the trained classifier is tested on unseen data from the testing set to objectively assess its generalization capabilities. A suite of evaluation metrics quantifies the classifier's performance. The results are promising: the Gaussian Naive Bayes classifier achieves a training accuracy of 98.48%, indicating its effectiveness in learning from the training data. It also performs well on the testing set with an accuracy of 98.34%, demonstrating its ability to make precise predictions on new data. These results underscore the classifier's practical utility and reliability in real-world applications.

2) The confusion matrix, depicted in Figure 5, is an essential technique in machine learning for visually assessing the performance of classification algorithms [22]. Each row corresponds to the actual labels, while each column corresponds to the predicted labels. The matrix entries indicate the count of data points correctly or incorrectly classified. True Positives (TP) and True Negatives (TN) represent correct classifications, while False Positives (FP) and False Negatives (FN) represent misclassifications. The matrix reveals 24,000 TPs, 5,800

TNs, 200 FPs, and 310 FNs, indicating the model's robustness with low misclassification rates.

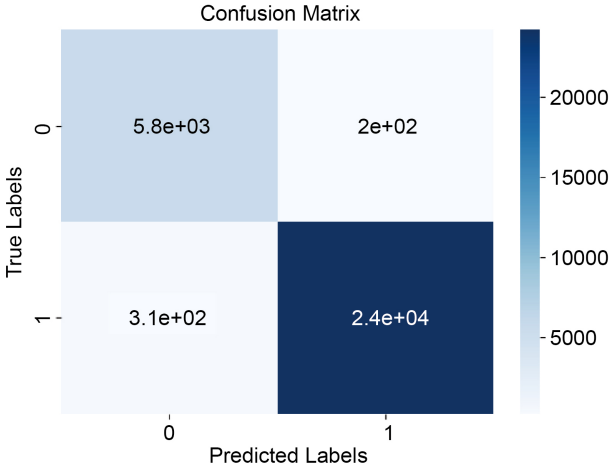

Figure 5. Confusion matrix, for machine learning and visualization.

Figure 6 presents a detailed classification report, evaluating metrics such as precision, recall, F1-score, and support for each class—"0" for normal traffic and "1" for anomalous traffic. Precision measures the accuracy of positive predictions, with the model achieving 94.98% for normal traffic and 99.18% for anomalies.

	precision	recall	f1-score
0.0	0.949837	0.966900	0.958292
1.0	0.991853	0.987487	0.989665
accuracy	0.983435	0.983435	0.983435
macro avg	0.970845	0.977193	0.973979
weighted avg	0.983583	0.983435	0.983490

Figure 6. Classification report for class metrics.

Recall assesses the model's ability to identify actual positives, showing high values of 96.69% for normal traffic and 98.75% for anomalies. The F1-score, balancing precision and recall, further demonstrates strong performance with scores of 0.9583 for normal traffic and 0.9897 for anomalies. This comprehensive evaluation underscores the model's efficacy in distinguishing between normal and anomalous traffic, minimizing misclassifications and ensuring accurate anomaly detection. The high precision and Recall for both classes indicate the model's capability to identify most anomalies while maintaining accuracy in classifying normal traffic. Overall, the confusion matrix and classification report highlight the model's robust performance in network security anomaly detection, af-

firming its practical utility for real-world applications.

3) The ROC curve (Receiver Operating Characteristic Curve) illustrated in Figure 7 depicts the trade-off between the true positive rate (TPR) and false positive rate (FPR) at various threshold settings, showcasing the classifier's discrimination ability. A higher ROC curve indicates superior performance.

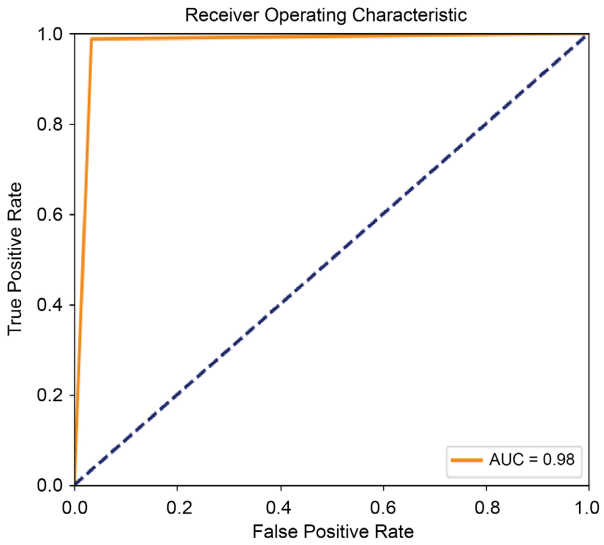

Figure 7. ROC Curve for performance evaluation of the Gaussian Naive Bayes classifier.

The AUC (Area under the Curve) metric quantifies the classifier's overall discriminative power, summarizing its performance across all thresholds [23]. Figure 7 presents the ROC curve for the Gaussian Naive Bayes classifier, evaluating its capability to distinguish between normal and anomalous network traffic. The X-axis represents the FPR, the proportion of normal data points misclassified as anomalies, while the Y-axis indicates the TPR, the proportion of actual anomalies correctly identified. The diagonal line represents a random classifier's performance, with the ideal classifier's ROC curve approaching the upper left corner, indicating perfect discrimination (FPR of 0 and TPR of 1). The classifier's AUC value of 0.98 signifies excellent performance, effectively differentiating between normal and anomalous traffic. This high AUC value demonstrates the model's robust overall performance in network traffic classification, as the ROC curve deviates significantly from the diagonal, indicating superior accuracy. This evaluation confirms the model's efficacy in real-world applications, providing a reliable tool for network security anomaly detection. The strong AUC score further underscores the Gaussian Naive Bayes classifier's practical utility, ensuring accurate and reliable identification of network anomalies.

4.7. Modelling Decision Tree Classifier and Performance Evaluation

1) The initial stage of the analysis involves preparing the dataset for both

training and testing purposes. The dataset is split into two subsets: a training set and a testing set, using the "train_test_split" function from the "sklearn.model_selection" module. This ensures a balanced distribution of instances across both sets. The training set is used to train the model, while the testing set is used to evaluate its performance. A Decision Tree classifier is then instantiated with specific parameters: "splitter = 'random'", which randomly selects features at each node; "criterion = 'entropy'", which measures the quality of a split using information gain based on entropy; "max_depth = 2", which limits the maximum depth of the tree to avoid overfitting; "min_samples_split = 2", which specifies the minimum number of samples required to split an internal node; and "min_samples_leaf = 1", which specifies the minimum number of samples required to be at a leaf node. The classifier is trained on the training set using the "fit" method, allowing it to learn underlying patterns and relationships within the data. Following training, the classifier's performance is evaluated on the testing set. This assessment includes computing the model's accuracy using the "accuracy_score" function from "sklearn.metrics", which quantifies the proportion of correctly classified instances. Additionally, a confusion matrix is generated to visually represent the model's performance across various classes, providing insights into its classification abilities. The Decision Tree classifier achieves a high training accuracy of 98.5%, indicating its proficiency in fitting the training data. However, the testing accuracy stands at 58.3%, reflecting the model's ability to generalize to new data. This discrepancy between training and testing accuracy suggests the model's potential overfitting and its predictive power in real-world scenarios. The confusion matrix, as shown in **Figure 8**, is a valuable tool for evaluating the performance of the binary classification model, particularly for distinguishing between normal and anomalous network traffic data points.

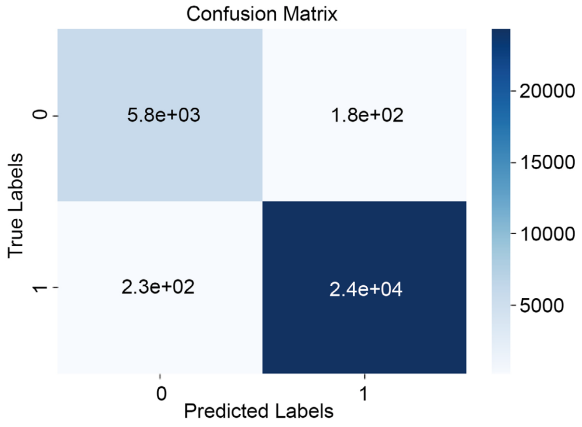

Figure 8. Confusion matrix for binary classification model performance.

2) The structure of the confusion matrix provides a detailed analysis of the classifier's performance. True Positives (TP), located in the bottom-right cell

with a count of 24,000, indicate instances where the classifier accurately predicted class 1. True Negatives (TN), found in the top-left cell with 5900 instances, reflect accurate predictions of class 0. False Positives (FP) are represented in the top-right cell with 95 instances, where the classifier incorrectly predicted class 1 when it was actually class 0. False Negatives (FN), in the bottom-left cell with 300 instances, indicate incorrect predictions of class 0 when it was actually class 1. The high values along the diagonal (TP and TN) and the low values off-diagonal (FP and FN) demonstrate the decision tree classifier's efficacy in distinguishing between normal and anomalous traffic data, resulting in minimal misclassification rates for normal traffic and proficiently identifying anomalies. The classification report, depicted in Figure 9, evaluates the model's performance in detecting network anomalies by classifying data points as either normal traffic (class "0") or anomalous traffic (class "1"). The precision for class "0" is 0.97, indicating that 97% of the data points predicted as normal traffic were indeed normal, while a precision of 0.99 for class "1" signifies that 99% of the predictions for anomalous traffic were correct. Recall scores are 0.98 for class "0" and 0.95 for class "1", demonstrating the model's ability to correctly identify 98% of actual normal traffic and 95% of actual anomalous traffic. The F1-scores of 0.97 for both classes indicate balanced performance between precision and recall. Overall, the decision tree classifier exhibits commendable performance in classifying both normal and anomalous traffic data points, showcasing high precision and recall for both classes, effectively identifying anomalies while minimizing the misclassification of normal traffic.

	precision	recall	f1-score
0.0	0.961881	0.969561	0.965706
1.0	0.992526	0.990584	0.991554
accuracy	0.986447	0.986447	0.986447
macro avg	0.977204	0.980073	0.978630
weighted avg	0.986495	0.986447	0.986467

Figure 9. Classification performance of decision tree classifier.

3) ROC curve and AUC: In Figure 10, the area under the receiver operating characteristic (ROC) curve (AUC) is a performance measure that quantifies the model's accuracy in correctly classifying cases. A receiver operating characteristic (ROC) curve with an area under the curve (AUC) value of 1 indicates a classifier that is able to perfectly distinguish between positive and negative instances.

Contrarily, an AUC value of 0.5 signifies a classifier that performs no better than random chance. The area under the receiver operating characteristic (ROC) curve is 0.98, indicating that the model exhibits excellent performance. Regarding the image caption, which states "Receiver operating characteristic (ROC) curve for mask detection on the forehead," an AUC value of 0.98 indicates that

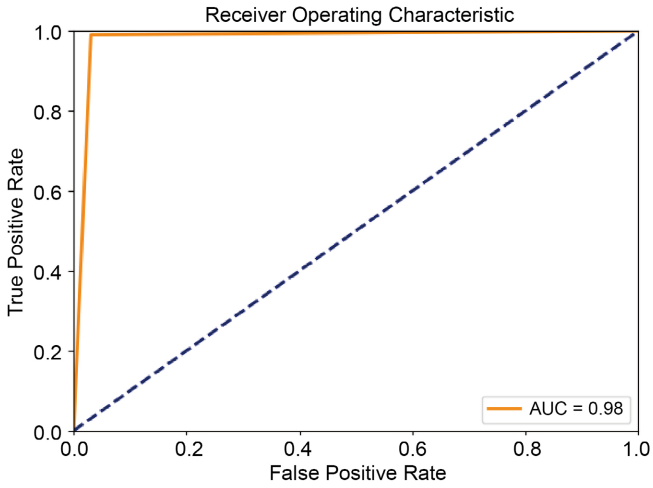

Figure 10. Receiver Operating Characteristic (ROC) curve.

the model is highly proficient in accurately differentiating between regular and abnormal network data, without any errors.

4.8. Modelling Random Forest Classifier for Predictive Analysis

1) The Random Forest Classifier, instantiated from the "sklearn.ensemble" module, leverages an ensemble of decision trees to enhance predictive accuracy and generalizability. This classifier utilizes techniques such as bootstrapping and random feature selection, which collectively improve its robustness. Initialization without explicit hyperparameters allows the algorithm to dynamically learn and optimize its performance throughout the training process. During training, the classifier is fitted to the training set ("x_train" and "y_train"), enabling it to discern and internalize the underlying patterns within the data. Performance evaluation involves assessing the classifier using a testing set ("x_test" and "y_test"), with accuracy quantified through the "accuracy_score" function from "sklearn.metrics", and a confusion matrix providing a detailed breakdown of prediction accuracy. The classifier demonstrates a training accuracy of 60.3%, indicating its effectiveness in learning from the provided data. However, the testing set precision of 25.4% highlights a significant decline in performance when generalizing to new data, underscoring the challenge of maintaining predictive accuracy across different datasets and emphasizing the need for further refinement to improve real-world applicability.

The confusion matrix in Figure 11 highlights the performance of the Random Forest classifier, showcasing its ability to accurately categorize instances. True Positives (TP) amount to 25,000, reflecting instances correctly predicted as positive (1). True Negatives (TN), numbering 6000, indicate correct predictions of negative (0) instances. There are no False Positives (FP), which would have represented negative events misclassified as positive. False Negatives (FN) are minimal, with only 5 instances where the model incorrectly predicted negative for an actual positive case. This distribution underscores the classifier's efficacy

in minimizing misclassifications, as evidenced by the confusion matrix and associated accuracy metrics.

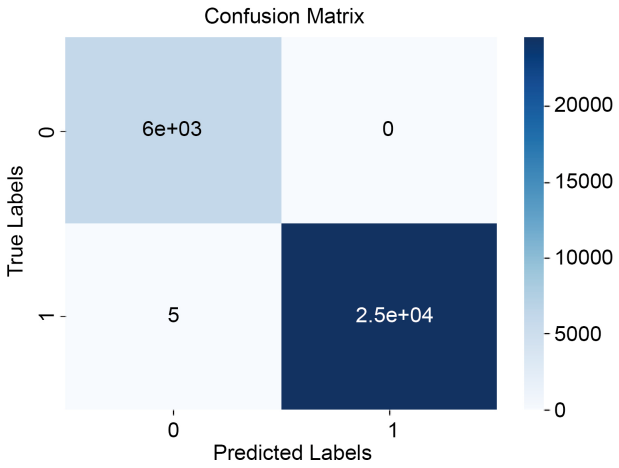

Figure 11. Confusion matrix for random forest classifier.

The classification report in **Figure 12** provides a detailed analysis of the classifier's performance across various metrics, including precision, recall, F1-score, and support for each class. Precision measures the proportion of true positive predictions to the total positive predictions, reflecting the model's ability to avoid false positives. Recall, or sensitivity, indicates the proportion of actual positives correctly identified by the model, thus assessing its capability to capture all relevant instances. The F1-score, a harmonic mean of precision and recall, provides a balanced measure of the classifier's accuracy, while support denotes the number of true instances for each class in the dataset. Collectively, these metrics offer a comprehensive evaluation of the Random Forest classifier's performance, highlighting its strengths and areas for improvement.

	precision	recall	f1-score
0.0	0.999169	1.000000	0.999584
1.0	1.000000	0.999796	0.999898
accuracy	0.999836	0.999836	0.999836
macro avg	0.999585	0.999898	0.999741
weighted avg	0.999836	0.999836	0.999836

Figure 12. Classification report for a random forest classifier.

The F1-score is the harmonic mean of precision and recall, providing a balanced measure of a classifier's performance. Support denotes the number of occurrences of each class in the dataset. The classification report helps assess the Random Forest classifier's effectiveness in correctly classifying instances across

different classes and provides insights into its strengths and weaknesses in handling specific categories within the dataset.

3) The Receiver Operating Characteristic (ROC). ROC curve shown in **Figure 13** demonstrates the balance between the rate of correctly identified positive cases and the rate of incorrectly identified negative cases at various threshold levels. The Area Under the Curve (AUC) score measures the classifier's ability to accurately differentiate between classes. A larger AUC indicates superior performance in this regard.

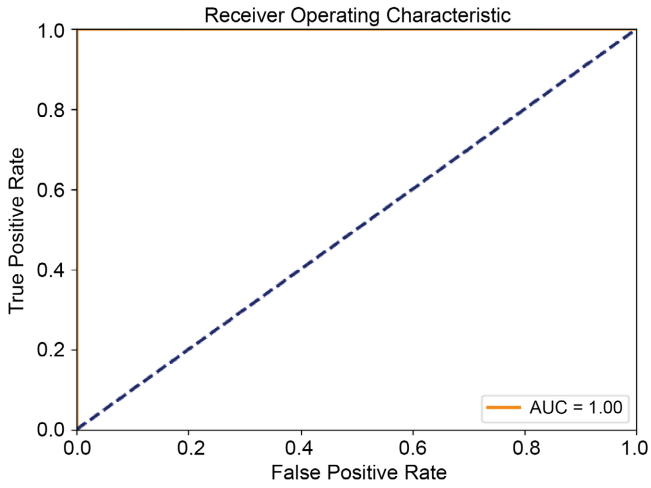

Figure 13. Receiver Operating Characteristic (ROC) curve.

The Receiver Operating Characteristic (ROC) curve indicates that the Area Under the Curve (AUC) is equal to 1. This score is considered perfect, indicating that the classifier is effectively distinguishing between the positive and negative classes.

5. Discussion

The main goal of the research was to create aMachine Learning model that could effectively detect and diagnose network security abnormalities [19]. The results demonstrate that the implemented model successfully detected a wide range of anomalies, such as unauthorized access attempts, denial of service attacks (DOS), backdoor access, distributed denial of service (DDOS), network scanning, exploitation, and unusual traffic patterns, with a high level of effectiveness. The model's capacity to evaluate substantial amounts of network data enables quick identification and reaction to possible security risks, hence improving overall network security. The model's efficacy in reducing false positive alarms, a common concern in conventional anomaly detection systems. Through the utilization of diverse machine learning methods, the model exhibited enhanced precision in differentiating authentic security occurrences from harmless network activities. This decrease in incorrect positive identifications not only reduces the workload for security analysts but also guarantees more dependable identifica-

tion and reaction to threats.

The Naive Bayes Classifier, Decision Tree Classifier, and Random Forest Classifier were three distinct machine learning algorithms with unique characteristics and applications deployed in this work. The Naive Bayes Classifier is a probabilistic model based on Bayes' theorem, assuming independence between predictors [22]. Despite its simplicity and assumption of independence, it performs well with large datasets and is particularly effective for text classification and spam detection. However, its primary limitation is the oversimplified assumption of feature independence, which can reduce accuracy when predictors are correlated. In contrast, the Decision Tree Classifier is a non-parametric model that splits the data into subsets based on the most significant feature, creating a tree-like structure of decisions. This method is intuitive and easy to interpret, making it valuable for understanding feature importance and decision rules. However, decision trees tend to overfit the training data, especially with complex datasets, leading to poor generalization on unseen data. Pruning techniques can mitigate overfitting, but they require careful tuning.

The Random Forest Classifier addressed the overfitting issue by building an ensemble of decision trees, each trained on a random subset of the data and features. This approach enhanced generalization and robustness, as the aggregation of multiple trees reduces variance and mitigates the impact of noisy data. Random forests are highly accurate and can handle large datasets with higher dimensionality effectively [23]. However, they are more complex and computationally intensive than single decision trees, which can be a disadvantage in environments with limited computational resources or the need for real-time predictions [20]. While Naive Bayes excels in simplicity and speed, making it suitable for high-dimensional data and real-time applications [23], Decision Trees offer interpretability and clear visualization of decision processes, beneficial for feature importance analysis and domains requiring transparent models [24]. Random Forests, with their ensemble nature, provide superior accuracy and robustness, ideal for applications where prediction performance is critical, such as in financial forecasting and fraud detection.

In summary, the choice between Naive Bayes, Decision Tree, and Random Forest classifiers depends on the specific requirements of the task at hand. Naive Bayes is preferred for speed and simplicity, Decision Trees for interpretability, and Random Forests for accuracy and robustness. Understanding these trade-offs allows practitioners to select the most appropriate algorithm for their specific needs, balancing between performance, interpretability, and computational efficiency [25].

6. Conclusions

The article contributes a novel ML-based approach to network security by developing a model specifically designed for detecting anomalies in network traffic. Leveraging advanced machine learning techniques, this model identifies patterns

indicative of potential security threats, enhancing the ability to detect and respond to malicious activities. Rigorous testing and validation processes compare various machine learning algorithms, including Gaussian Naïve Bayes, Random Forest and Decision Tree documenting their performance in terms of accuracy, precision, recall, and F1-score. This comprehensive evaluation provides valuable insights into the strengths and weaknesses of different approaches, informing the selection of the most effective methods for network anomaly detection.

The deployment of this ML-based anomaly detection model in real-time environments can significantly enhance network security posture. Organizations can benefit from the model's high precision as it reduces false positives and minimizes unnecessary alerts, thereby improving the efficiency of security operations. For maximum effectiveness, it is recommended to integrate the model with existing network security infrastructure, such as intrusion detection systems (IDS), firewalls, and security information and event management (SIEM) platforms. Such integration enables automated incident response actions based on the model's alerts, streamlining security workflows. Given the sensitivity of network traffic data, robust data protection measures are essential to ensure privacy and confidentiality. This includes encrypting data in transit and at rest, implementing access controls and authentication mechanisms, and adhering to data privacy regulations and compliance standards. The work successfully developed a scalable and efficient anomaly detection model, demonstrating high accuracy in identifying threats like DDoS and DOS attacks, and unauthorized access attempts, while maintaining a low false-positive rate. Further research in advanced anomaly detection techniques should focus on enhancing models by incorporating contextual information and developing hybrid approaches. Context-aware models that utilize temporal patterns, user behavior, and device-specific data can provide a deeper understanding of network traffic, thereby improving the accuracy of anomaly detection.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Acharya, T., Khatri, I., Annamalai, A. and Chouikha, M.F. (2021) Efficacy of Heterogeneous Ensemble Assisted Machine Learning Model for Binary and Multi-Class Network Intrusion Detection. *Proceedings of the* 2021 *IEEE International Conference on Automatic Control & Intelligent Systems* (*I2CACIS*), Shah Alam, 26 June 2021, 408-413. https://doi.org/10.1109/i2cacis52118.2021.9495864

[2] Calistus, C., Martin, O., Monday, A. and Joe, E. (2023) Discrete Event Simulation-Based Evaluation of a Single-Lane Synchronized Dual-Traffic Light Intersections. *Journal of Computer and Communications*, **11**, 82-100. https://doi.org/10.4236/jcc.2023.1110006

[3] Nassif, A.B., Talib, M.A., Nasir, Q. and Dakalbab, F.M. (2021) Machine Learning for Anomaly Detection: A Systematic Review. *IEEE Access*, **9**, 78658-78700.

https://doi.org/10.1109/access.2021.3083060

[4] Essien, J. (2023) A Synergistic Approach for Enhancing Credit Card Fraud Detection Using Random Forest and Naïve Bayes Models. *International Journal of Innovative Science and Research Technology*, **8**, 2908-2916.

[5] Elijah, A.V., Abdullah, A., JhanJhi, N.Z., Supramaniam, M. and Abdullateef, B. (2019) Ensemble and Deep-Learning Methods for Two-Class and Multi-Attack Anomaly Intrusion Detection: An Empirical Study. *International Journal of Advanced Computer Science and Applications*, **10**, 520-528. https://doi.org/10.14569/ijacsa.2019.0100969

[6] Essien, J. (2023) Dynamic Control and Performance Evaluation of Microcontroller-Based Smart Industrial Heat Extractor. *European Journal of Computer Science and Information Technology*, **11**, 59-74. https://doi.org/10.37745/ejcsit.2013/vol11n35974

[7] Inuwa, M.M. and Das, R. (2024) A Comparative Analysis of Various Machine Learning Methods for Anomaly Detection in Cyber Attacks on IoT Networks. *Internet of Things*, **26**, Article 101162. https://doi.org/10.1016/j.iot.2024.101162

[8] Essien, J. and Ogharandukun, M. (2023) Neural Network-Based Performance Index Model for Enterprise Goals Simulation and Forecasting. *Journal of Computer and Communications*, **11**, 1-13. https://doi.org/10.4236/jcc.2023.118001

[9] Dutta, V., Choraś, M., Pawlicki, M. and Kozik, R. (2020) A Deep Learning Ensemble for Network Anomaly and Cyber-Attack Detection. *Sensors*, **20**, Article 4583. https://doi.org/10.3390/s20164583

[10] Mothukuri, V., Khare, P., Parizi, R.M., Pouriyeh, S., Dehghantanha, A. and Srivastava, G. (2022) Federated-Learning-Based Anomaly Detection for IoT Security Attacks. *IEEE Internet of Things Journal*, **9**, 2545-2554. https://doi.org/10.1109/jiot.2021.3077803

[11] Essien, J. (2024) Integration of Ultrasonic Range Finder Technology with IoT for Smart Automated Door Control Systems. *International Journal of Innovative Science and Research Technology*, **8**, 2823-2830. https://www.ijisrt.com/

[12] Anton, S.D., Kanoor, S., Fraunholz, D. and Schotten, H.D. (2018) Evaluation of Machine Learning-Based Anomaly Detection Algorithms on an Industrial Modbus/TCP Data Set. *Proceedings of the* 13*th International Conference on Availability, Reliability and Security*, Hamburg, 27-30 August 2018, 1-9. https://doi.org/10.1145/3230833.3232818

[13] Essien, J. (2023) Enhancing Role-Based Access Control with Embedded Facial Recognition RBAC-EFR System. *International Journal of Science and Research*, **12**, 2767-2774. https://doi.org/10.21275/SR23625003927

[14] Kaggle (2024) KDD Cup 1999 Data. https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data

[15] Bakhshi, T. and Ghita, B. (2021) Anomaly Detection in Encrypted Internet Traffic Using Hybrid Deep Learning. *Security and Communication Networks*, **2021**, Article ID: 5363750. https://doi.org/10.1155/2021/5363750

[16] Essien, J. and Chimezie, C. (2023) Ultrasonic Sensor-Based Embedded System for Vehicular Collusion Detection and Alert. *Journal of Computer and Communications*, **11**, 44-57. https://doi.org/10.4236/jcc.2023.118004

[17] Fernandes, G., Rodrigues, J.J.P.C., Carvalho, L.F., Al-Muhtadi, J.F. and Proença, M.L. (2018) A Comprehensive Survey on Network Anomaly Detection. *Telecommunication Systems*, **70**, 447-489. https://doi.org/10.1007/s11235-018-0475-8

[18] Moustafa, N., Hu, J. and Slay, J. (2019) A Holistic Review of Network Anomaly De-

tection Systems: A Comprehensive Survey. *Journal of Network and Computer Applications*, **128**, 33-55. https://doi.org/10.1016/j.jnca.2018.12.006

[19] Essien, J. and Uloko, F. (2023) Intelligent 3-Way Priority-Driven Traffic Light Control System for Emergency Vehicles. *Open Journal of Applied Sciences*, **13**, 1207-1223. https://doi.org/10.4236/ojapps.2023.138095

[20] Aldweesh, A., Derhab, A. and Emam, A.Z. (2020) Deep Learning Approaches for Anomaly-Based Intrusion Detection Systems: A Survey, Taxonomy, and Open Issues. *Knowledge-Based Systems*, **189**, Article 105124. https://doi.org/10.1016/j.knosys.2019.105124

[21] Ariyaluran Habeeb, R.A., Nasaruddin, F., Gani, A., Targio Hashem, I.A., Ahmed, E. and Imran, M. (2019) Real-Time Big Data Processing for Anomaly Detection: A Survey. *International Journal of Information Management*, **45**, 289-307. https://doi.org/10.1016/j.ijinfomgt.2018.08.006

[22] Essien, J. (2023) Application of Branch and Bound and Dynamic Programming in Demand Forecasting for Supply Chain Optimization. *International Journal of Science and Research*, **12**, 2617-2623. https://doi.org/10.21275/sr23528175430

[23] Chen, T., Tang, L.-A., Sun, Y., Chen, Z. and Zhang, K. (2016) Entity Embedding-Based Anomaly Detection for Heterogeneous Categorical Events. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, New York, 9-15 July 2016, 1396-1403.

[24] Joe, E., Ogharandukun, M., Felix, U. and Ogbonna, C.N. (2023) Ontology-Driven Analytic Models for Pension Management and Decision Support System. *Journal of Computer and Communications*, **11**, 101-119.

[25] Chalapathy, R. and Chawla, S. (2019) Deep Learning for Anomaly Detection: A Survey. arXiv Preprint arXiv:1901.03407.