Anomaly Detection : Unusual Activities in Network Activity

*Note: Sub-titles are not captured in Xplore and should not be used

Madeline Andrea Sofian
Computer Science Department

School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
madeline.sofian@binus.ac.id

Angel Priscilla Salim
Computer Science Department

School of Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
angel.salim@binus.ac.id

***Abstract*—This electronic document is a "live" template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. **CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.* (*Abstract*)**

Keywords—component, formatting, style, styling, insert (*key words****)***

I. Introduction

The rapid growth of digital infrastructure empowered and transformed how people communicate, and data transferred across. Concurrently, there has been a significant rise in the amount of network activities, both benign and malicious, underscoring the imperative for stronger network security. The rise of digital networks consequently leads to a rise in the number of potential network intrusions and security breaches [1] Attacks can vary from unauthorized access, data exfiltration to more specific attacks such as the distributed denial-of-service (DDoS) attacks.

Hence, early detection in unusual network activities is fundamental in ensuring system integrity and safeguarding sensitive information [2]. As networks become more complex, traditional security measures often struggle to adapt, it is imminent to develop more sophisticated techniques to detect unusual patterns in network traffic activities. The application of Artificial Intelligence (AI) in enhancing anomaly detection systems has proven to be a breakthrough, significantly improving the detection of unusual network activities [3]. AI-based methods can analyze vast amounts of network traffic data in real-time, identifying subtle patterns and anomalies that traditional systems may overlook.

The objective from the data mining perspective aims to identify critical elements in network traffic that are associated with malicious activity, such as packet size, source IP address, and protocol type. These factors help provide insights into the factors that determine whether network traffic is considered benign or malicious and evaluate the effectiveness of algorithms for detecting such anomalies by further studying these characteristics. In addition, a robust anomaly detection system can significantly mitigate the costs of cyberattacks, including operational disruptions and reputational damages of businesses. By utilizing their optimal algorithm to detect and prioritize actions, this helps ensure the protection of sensitive data and uninterrupted business continuity.

This research will evaluate and compare three anomaly detection algorithms being Isolation Forest, XGBoost, and Local Outlier Factor (LOF). These algorithms will be trained using 2 types of datasets that will be merged which are the "UNSW-NB15" dataset comprising of 700,001 data entries labeled as "1" (Benign) and "2" (Malicious) and the "Malware Detection in Network Traffic Data" dataset comprising of 1,008,748 data entries labeled as "Benign" and "Malicious". Through thorough training and evaluation of these algorithms, this research aims to create a scalable and robust solution that can effectively assist stakeholders in detecting anomalous network activities early and as a result, aid in protecting their business interests.

II. Theoretical Basis

Anomaly detection plays a critical role in network traffic analysis by identifying patterns or behaviors that deviate from the expected norm. Such anomalies often indicate critical security issues, including unauthorized access, distributed denial-of-service (DDoS) attacks, or the spread of malware[4]. These anomalies fall into three main categories: point anomalies, which occur when individual data points deviate significantly from the norm; contextual anomalies, which occur when deviations are context-dependent and collective anomalies, which occur when a group of data points collectively signifies abnormal behavior. The dynamic and high-dimensional nature of network traffic introduces challenges such as scalability, real-time detection, and the difficulty of obtaining labeled anomaly data, necessitating the use of efficient and robust anomaly detection algorithms.

Machine learning serves as a fundamental tool in anomaly detection, with methodologies broadly categorized into supervised, semi-supervised, and unsupervised learning. Supervised learning techniques rely on labeled dataset, where both normal and anomalous instances are identified, enabling the model to learn explicit patterns [5]. The availability of labeled datasets in real-world network environments is limited. Semi-supervised learning approaches, which use datasets containing only normal instances to identify deviations, and unsupervised learning approaches, which require no labeled data are more practical

for large-scale network traffic. These methods are particularly suited for dynamic environments where anomalies evolve over time, making them invaluable for real-time anomaly detection tasks.

A. XGBoost

XGBoost or Extreme Gradient Boosting is a supervised learning algorithm widely used for both classification and regression tasks. It is based on the principle of gradient boosting, which sequentially builds an ensemble of decision trees to minimize a regularized loss function[6]. Unlike traditional gradient boosting, XGBoost incorporates optimizations such as sparsity awareness, parallel processing, and advanced regularization techniques, making it faster and more efficient. The objective function in XGBoost calculates the total of the loss function and the regularization term controlling model complexity. XGBoost is particularly effective in handling imbalanced datasets, missing data, and complex feature interactions, making it suitable for anomaly classification tasks. On the other hand, it requires labeled data, which may not always be available in real-world anomaly detection scenarios.

B. Isolation Forest

Isolation Forest algorithm is an unsupervised anomaly detection technique designed to isolate anomalies in a dataset. The key concept is that anomalies are easier to isolate than regular data points due to their rarity and dissimilarity. The algorithms construct a series of binary trees (isolation trees) by randomly partitioning the dataset along feature dimensions. [7] The depth of a data point in the tree is used to compute its anomaly score. The average path lengths within the trees are often shorter for anomalies, which are more distinct and isolated. The anomaly score for a point is computed as 2 raised to the power of negative expected path length of a point in the isolation tree, divided by the average path length of unsuccessful searches in a binary tree. Anomalies are shown by scores near 1, and normal data points are represented by scores around 0. Advantages of Isolation Forest include its computational efficiency, scalability to high-dimensional data, and minimal assumptions of data distribution. On the other hand, it assumes that anomalies are sparse and distinct, reducing its effectiveness in datasets with subtle or overlapping anomalies.

C. Local Outlier Factor (LOF)

Local Outlier Factor (LOF) is an unsupervised anomaly detection method that identifies data points with significantly lower density than their neighbors. LOF uses the basic concept of local reachability density (LRD), which measures the density of a point relative to its *k*-nearest neighbors [8]. The LOF score is computed as the ratio of the average LRD of its neighbors to its own LRD. A score significantly greater than 1 indicates an anomaly. LOF is highly effective in identifying localized anomalies in datasets with varying densities. On the other hand, its performance on large-scale datasets are affected by its computational requirements and the choice of *k*, the number of neighbors.

III. METHODOLOGY

A. Datasets

This first dataset named “UNSW-NB15” was taken from Kaggle comprising of 49 attributes and 700,001 labeled data entries being either “1” (Benign) and “2” (Malicious). This second dataset named “Malware Detection in Network Traffic Data” was also taken from Kaggle comprising of 23 attributes and 1,008,748 data entries labeled as “Benign” and “Malicious”. Both datasets were merged based on 15 common attributes namely “SourceIP”, “Protocol”, “State”, and “Service” and were eventually split into 80% for the training dataset and 20% for the testing dataset.

B. Methods

1. XGBoost

   XGBoost was selected for this research due to its robustness and efficiency in handling sparse data and optimized computation. Moreover, its ability to prune trees during the training process and conduct parallel processing makes training less time consuming. XGBoost is also well known for its highly customizable objective function allowing for meticulous control over the learning process which proves to be effective in detecting anomalies in highly dimensional data.

2. Isolation Forest

   Isolation Forest was selected for this research due to its suitability to work with high-dimensional data and imbalanced data. It is a well-known fact that in a network activity dataset, most data entries would be labeled as “Benign” as compared to “Malicious”. Additionally, Isolation Forest is well-suited for handling imbalanced datasets as it does not rely on class distributions assumptions but instead isolates anomalies based on rarity and uniqueness.

3. Local Outlier Factor (LOF)

   LOF was selected for this research due to its ability to provide detailed detection of anomalies in localized areas. This feature makes the identification of subtle deviations in network patterns to be more suitable and effective. Moreover, comparing the local density of a point to that of its neighbors makes LOF particularly effective for detecting anomalies in datasets where density varies across regions.

C. Exploratory Data Analysis

1. Data Preprocessing

   This step involves pre-processing the merged dataset to prepare it for EDA and ensure data consistency across all the data entries. Initially we identify common columns between both datasets manually through further data understanding of each of their attributes. Based on our analysis, we discovered that there were 15 common attributes being “ID”, “TimeStamp”, “SourceIP”, “SourcePort”, “DestinationIP”, “DestinationPort”, “Protocol”, “Service”, “Duration”, ‘BytesSent”, “BytesReceived”, “State”, “PacketsSent”,

"PacketsReceived" and "Label". The datasets were then integrated based on these common columns and removed duplicated data where necessary.

Since missing values in the dataset varies between NA and "-", we standardized the type to NA to aid in handling for missing values. For categorical attributes that have less than 10% composition of missing values such as "SourcePort" and "DestinationPort", we replaced the missing values with the mode of each attribute. On the other hand, for attributes that have more than 10% composition of missing values were removed as their significant number of missing values might provide inaccurate findings.

To select features that are significant to the network activity, the correlation heatmap was plotted. The correlation heatmap highlights the factors that influence whether the network activity is normal (Benign) or unusual (Malicious) as shown in the Fig.1. below.

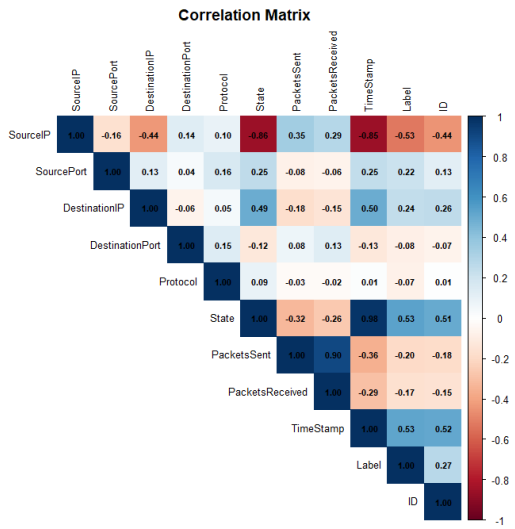

Fig.1. Correlation heatmap

Stronger correlations are identified based on larger nominal values and darker colours. Attributes that have a positive correlation value are directly related to the Malicious label whereas those that have a negative correlation value are inversely related to the Malicious label. Based on the above figure, attributes that show strong correlation to the Label column are "State" (of connection) with positive value of "0.53", "TimeStamp" with positive value of "0.53" and "SourceIP" with negative value of "0.53". Hence this implies that "State" and "TimeStamp" have a strong direct relation whereas the "SourceIP" has a strong inverse relation with the Malicious Label.

Additionally, attributes with weaker correlations such as the "SourcePort", "DestinationIP" and (Amount of) "Packets Sent" have values "0.22", "0.24" and "-0.20" respectively. Though weaker, these factors are still worth analysing for since they have values that fall more than or equal to the feature selection threshold which is |0.20|.

2. Plotting and Further Analyses

To further understand the relationships between the selected features and the target variable (Malicious Labels), we conducted further analyses following the initial correlation heatmap. The features chosen for deeper exploration—TimeStamp, State (of connection), Packets Sent, DestinationIP, SourceIP, and SourcePort—were identified as potentially significant in distinguishing between benign and malicious activities.

For each feature, we visualized its distribution and interaction with the Malicious Labels column using various plots. Temporal patterns were analyzed for "TimeStamp" to identify trends or irregularities over time as plotted in Fig.2. as a line chart.

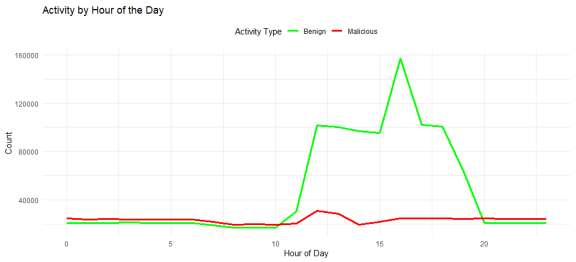

Fig.2. Temporal Pattern of Network Activity against Time (Hour) within 24 hours.

Based on the plot, the number of unusual activities or potential attacks usually rise during the noon at about 12 to 1 pm from about 10,000 to 20,000 counts of malicious labeled activities.

The "State" feature was further examined to assess its relation to network activity status where we were able to identify at which states attacks are usually launched. This is plotted using a stacked bar chart as shown in Fig.3.

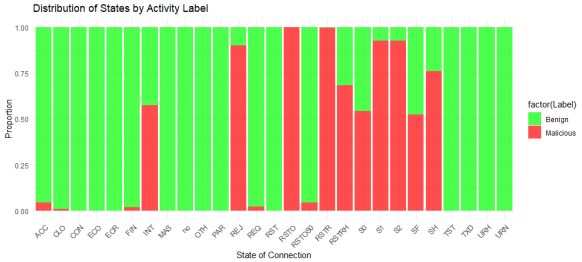

Fig.3. Proportion of States of Connection used for malicious activities.

Based on the plot, unusual activities/attack usually occur during the INT, REJ, RSTD, RSTR, RSTRH, S0, S1, S2, SF, SH connection state with a more than or equal to "0.50" proportion of the entire network activity.

For numeric features like (number of) “PacketsSent”, comparisons between benign and malicious traffic were performed to indicate the common number of packets sent from the Source during an attack. To visualize this, a stacked bar chart was plotted in Fig.4.

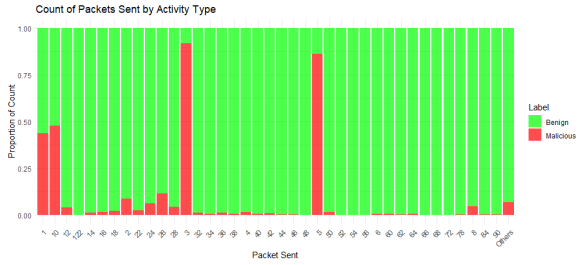

Fig.4. A proportion of number of packets sent as malicious activities.

Based on the plot, unusual activities/attack generally send packets with length 1, 3, 5 and 10 as this makes up about more than or equal to “0.50” of the entire proportion of network activity.

For categorical features such as “DestinationIP”, we further analysed the Destination IPs mostly targeted to conduct unusual activities. This is visualized using a bar chart as shown in Fig.5.

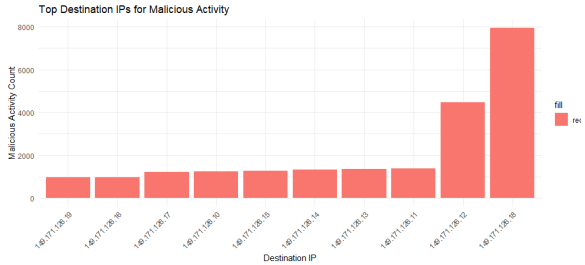

Fig.5. Destination IPs commonly targeted for malicious activities.

Based on the plot, unusual activities/attack are mostly conducted target Destination IP 149.171.126.18 and 149.171.126.12 with malicious activity of over 4000 counts.

Similary, for “SourceIP” we also performed further analysis on the Source IP mostly used to conduct unusual network activities. This is visualized using a bar chart as shown in Fig.6.

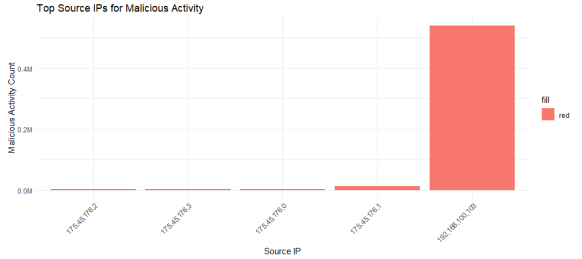

Fig.6. Source IPs commonly used for malicious activities.

Based on the plot, unusual activities/attack are mostly conducted by Source IP 192.168.100.103 with malicious activity count amounting to nearly “0.55M”.

For “SourcePort”, we analyzed using stacked bar plots to uncover any distinct patterns associated with malicious behavior. Based on this analysis, we were able to highlight the Source Port numbers that are frequently used to conduct unusual activities/attack as seen in Fig.7.

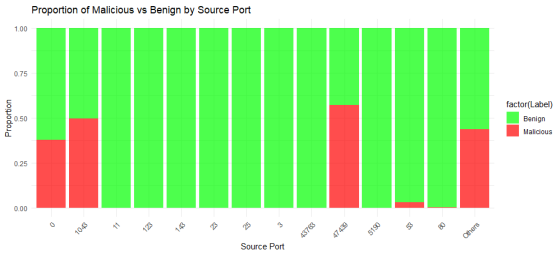

Fig.7. Source Ports commonly used for malicious activities.

Based on the plot, unusual activities/attack are mostly conducted by Source Port, 1043 and 47439 as they make up more than or equal to “0.50” of the entire proportion of network activities.

D. *Training Procedure*

1. Data Preprocessing

This step involves pre-processing the merged dataset to prepare it for training and ensure data consistency across all the data entries. The steps for preprocessing are generally similar to the one conducted for EDA which involved removal of duplicated data, aligning common attributes, data integration, handling of missing values and feature selection. Additionally, the merged dataset was split into 80% training and 20% testing set using stratifies sampling to preserve label distribution.

2. Model Initialization & Training

The XGBoost model was trained using hyperparameters such as max depth of 5, lambda of 1, alpha of 1 and nrounds of 100. Early stopping was employed to stop training after 10 rounds of no improvement.

The LOF model was trained using parameter neighbors set to 20 which represents the number of neighbors used to calculate the local density. The threshold was set to the 95th percentile of LOF scores to classify anomalies in the training set.

The Isolation Forest model was trained using the number of trees set to 100 and sample size of 480. The threshold was also set to the 95th percentile of Isolation Forest scores to classify anomalies in the training set.

3. Evaluation

Models	Specificity	Sensitivity	Accuracy

XGBoost	0.9988	0.9994	0.9992
Isolation Forest	0.08219	0.9641	0.6958
LOF	0.1128	0.9537	0.7175

Fig.8. Tables of summary of the 3 models

Based on Fig.8., XGBoost has the highest Specificity value of "0.9988" as compared to Isolation Forest with "0.08219" and LOF with "0.1128" implying that XGBoost has a better ability in identifying negative cases correctly. XGBoost also outperformed both other 2 models in terms of sensitivity scoring "0.9994" where Isolation Forest scored "0.9641" and LOF scored "0.9537". This means that XGBoost performed best in correctly identifying positive cases though Isolation Forest and LOF performed well as well. Lastly, XGBoost had a higher accuracy of "0.9992" as compared to Isolation Forest and LOF with "0.6958" and "0.7175" respectively. Hence in terms of performance overall, XGBoost performed best when predicting for both benign and malicious activities due to having the highest accuracy value. This evaluation will help in identifying the most suitable model for performing anomaly detection in network traffic more accurately.

IV. Results

Isolation Forest achieves an overall accuracy of 69.58%, meaning that approximately 69.58% of the model's predictions align with the true labels. It achieves 96.47% on sensitivity (recall), indicating that the model is highly effective at identifying anomalous traffic. On the other hand, Isolation Forest only achieves 8.21% on specificity, indicating that the model misclassifies a large proportion of normal traffic as anomalies. A low number of specificities suggests that the model may be overly sensitive to deviations from the norm, resulting in a high rate of false positives.

The performance of Isolation Forest can be influenced by several factors. First, Isolation Forest is an unsupervised algorithm that isolates outliers by partitioning the feature space using random splits. Its high sensitivity indicates that the anomalies in the dataset have distinct patterns or deviations that the algorithm can isolate effectively. However, its tendency to over-isolate leads to the misclassification of normal traffic as anomalies, as reflected in its low specificity. Second, Isolation Forest operates without explicit knowledge of class labels, which limits its ability to distinguish between normal and anomalous traffic accurately. It can lead to suboptimal classification performance when the dataset exhibits complex structures or overlapping distributions. Third, the performance of Isolation Forest is highly sensitive to parameters, such as the number of trees and the subsampling size. Suboptimal choices for these parameters may contribute to the model's over-prediction of anomalies and its inability to balance sensitivity and specificity effectively.

Local Outlier Factor (LOF) achieves an overall accuracy of 71.75%, meaning that approximately 71.75% of the model's prediction aligns with the actual label. It achieves 95.37% on sensitivity, indicating that the model correctly identifies a very high proportion of anomalous traffic. However, the model only achieves 11.28% on specificity, meaning that it fails to correctly classify the majority of normal traffic and leading to a high number of false positives. The imbalance indicates that the model is heavily biased towards detecting anomalies.

The performance of LOF heavily depends on data distribution, feature scaling, and hyperparameter tuning. LOF algorithms is designed to detect data points that deviate significantly from local density patterns. The high sensitivity indicates that the anomalies in the dataset exhibit clear deviations from normal traffic patterns, enabling the model to identify them effectively. The low specificity indicates that the normal traffic may exhibit a high degree of variability or overlap with anomalous patterns, leading to model's difficulty in differentiating between the two. Additionally, the unsupervised nature of LOF means that it lacks explicit knowledge of class labels, which may contribute to its poor performance in normal traffic classification. The dataset contains inherent properties, such as overlapping features, noisy data, or insufficient feature engineering, which leads to model's inability to achieve a better balance between sensitivity and specificity. Additionally, the model's prediction may be distorted due to anomalies predominating the dataset.

XGBoost achieves the highest overall accuracy of 99.92% among the three models. It achieves 99.94% on sensitivity, reflecting the model's ability to correctly identify almost all instances of anomalous traffic. The model achieves 99.88% on specificity, indicating that the model is highly effective in correctly classifying normal traffic and minimizing false positives. These numbers indicate the model's capacity to handle both classes with minimal errors.

Several factor contribute to the performance of XGBoost model. XGBoost is a gradient-boosting algorithm that excels in handling large-scale and high-dimensional data. Its ensemble-based approach minimizes overfitting by combining weak learners to achieve strong predictive capabilities. The model's ability to capture non-linear relationships and interactions between features is particularly advantageous in anomaly detection, where complex patterns often exist. Additionally, careful hyperparameter tuning, such as optimizing learning rate, tree depth, and regularization parameters contributed to the model's strong generalization ability. The performance of the model was further enhanced by XGBoost's built-in regularization mechanisms, such as L1 and L2 penalties to help control overfitting.

Code Link : https://drive.google.com/drive/folders/1-M7PZP-7ECw_NBGk4nRvOEcEGBbbZt1b?usp=sharing

V. Conclusion

In conclusion, this research focused on performing anomaly detection in network activities by comparing 3 different models that are suited in identifying malicious behavior within network traffic. Among the 3 models trained and tested, the XGBoost model outperformed the 2 other models and best distinguished between benign and malicious instances, achieving promising results in minimizing false positives while detecting anomalous patterns. However, future improvements could include refining feature selection, incorporating more real-life datasets, addressing class imbalances more effectively, and exploring other machine learning models for better

performance comparison. This will help create a more relevant and in-depth analysis of the real-life scenarios of network attacks. Further work will focus on enhancing model robustness, real-time anomaly detection, and integrating more advanced techniques to improve both detection accuracy and efficiency in network security applications.

REFERENCES

[1] X. Sun, "The Current Status and Challenges of Cybersecurity Risks," *Internet of Things and Cloud Computing*, vol. 12, no. 1, pp. 10–16, Jul. 2024, doi: 10.11648/j.iotcc.20241201.12.

[2] A. R. Yeruva, P. Chaturvedi, A. L. N. Rao, S. C. DimriL, C. Shekar, and B. Yirga, "Anomaly Detection System using ML Classification Algorithm for Network Security," in *2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, IEEE, Dec. 2022, pp. 1416–1422. doi: 10.1109/IC3I56241.2022.10072303.

[3] Vishnu Priya P M and Soumya S, "Advancements in Anomaly Detection Techniques in Network Traffic: The Role of Artificial Intelligence and Machine Learning," *Journal of Scientific Research and Technology*, pp. 38–48, Jun. 2024, doi: 10.61808/jsrt114.

[4] A. Ahmed, S. Hameed, M. Rafi, and Q. K. A. Mirza, "An Intelligent and Time-Efficient DDoS Identification Framework for Real-Time Enterprise Networks: SAD-F: Spark Based Anomaly Detection Framework," *IEEE Access*, vol. 8, pp. 219483–219502, 2020, doi: 10.1109/ACCESS.2020.3042905.

[5] W. Xu, "Advancements in Machine Learning for Network Anomaly Detection: A Comprehensive Investigation," in *Proceedings of the 1st International Conference on Engineering Management, Information Technology and Intelligence*, SCITEPRESS - Science and Technology Publications, 2024, pp. 585–589. doi: 10.5220/0012959700004508.

[6] Z. Arif Ali, Z. H. Abduljabbar, H. A. Tahir, A. Bibo Sallow, and S. M. Almufti, "eXtreme Gradient Boosting Algorithm with Machine Learning: a Review," *Academic Journal of Nawroz University*, vol. 12, no. 2, pp. 320–334, May 2023, doi: 10.25007/ajnu.v12n2a1612.

[7] M. H. Krishna, N. K, G. Charmitha, T. Vignesh, V. Ch, and S. Kuchibhotla, "Studies on Anomaly Detection Techniques," in *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*, IEEE, Feb. 2023, pp. 813–817. doi: 10.1109/ICCMC56507.2023.10083885.

[8] A. Wahid and A. Chandra Sekhara Rao, "An Outlier Detection Algorithm based on KNN-kernel Density Estimation," in *2020 International Joint Conference on Neural Networks (IJCNN)*, IEEE, Jul. 2020, pp. 1–8. doi: 10.1109/IJCNN48605.2020.9207033.