International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 21 No. 3 (1) 2025

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, No. 3(1), 2025

Research Paper

EDGE-ENABLED MACHINE LEARNING FRAMEWORK FOR REAL-TIME ANOMALY DETECTION IN IOT NETWORK

P. Satish[1], Ch. Venu Yadav[2], V. Raj Kumar[2], B. Sai Krishna Reddy[2], T. Vamshi Krishna[2]
[1]Assistant Professor, [2]UG Student, [1,2]Department of Computer Science and Engineering (Cyber Security),
[1,2]Sree Dattha Group of Institutions, Sheriguda, Ibrahimpatnam, 501510, Telangana

ABSTRACT

With the rapid expansion of IoT networks, global internet-connected devices are projected to exceed 29 billion by 2030, creating unprecedented volumes of real-time traffic. Nearly 70% of IoT devices are vulnerable to at least one security threat, and over 60% of reported anomalies go undetected due to inadequate early-warning mechanisms. The economic impact of network anomalies is significant, with businesses losing an estimated $120 billion annually due to undetected cyber threats, system downtime, and performance degradation. Traditional manual anomaly detection methods, such as signature-based identification, threshold monitoring, and log inspection, are increasingly ineffective in dynamic IoT environments. These techniques are time-intensive, prone to human error, and incapable of detecting zero-day anomalies or adapting to evolving traffic behaviors. To overcome these limitations, the proposed method presents a machine learning-driven anomaly detection framework tailored for IoT edge devices. The system begins with an end-to-end preprocessing pipeline that includes structured data exploration, class balance visualization, and feature standardization to optimize learning efficiency. Logistic Regression and an AdaBoost-enhanced Decision Tree Classifier are employed to classify four critical network anomaly types: Frequency Drift, Capacity Breach, Dual Signal Interference, and Request Overload. A performance evaluation module computes key metrics such as accuracy, precision, recall, and F1-score, and uses confusion matrices for interpretability. The architecture also supports real-time predictions on new data inputs, making the system practical for deployment in IoT-enabled infrastructures. By combining model reusability with scalable preprocessing and automated classification, this method enhances anomaly detection reliability and responsiveness in modern edge environments.

Received: 14-7-2025 Accepted: 21-8-2025 Published: 28-8-2025

1. INTRODUCTION

The exponential increase in internet-connected devices, cloud-based services, and digital communication has significantly expanded the scale and complexity of network traffic. According to Cisco's Annual Internet Report, global IP traffic is expected to reach 396 exabytes per month by 2025, up from 122 exabytes per month in 2020. This unprecedented growth raises considerable challenges in securing networks, managing bandwidth, and detecting abnormal or malicious activities. Anomalies in network traffic can indicate security breaches, operational failures, or even cyberattacks such as DDoS or insider threats, making real-time detection critical.Traditional rule-based systems struggle to keep up with the velocity and variability of modern traffic patterns, especially when adversaries employ stealthy or adaptive tactics. Statistical models and machine learning methods have been widely adopted to classify, cluster, or detect anomalies in traffic streams, leveraging packet headers, flow features, and session metadata. In particular, methods like Principal Component Analysis (PCA), k-means, and Isolation Forests have been widely used. However, these models often fail to provide scalable and distributed solutions for high-dimensional, real-time data environments where latency

and accuracy must be balanced. Recent advancements in optimization and distributed learning frameworks have

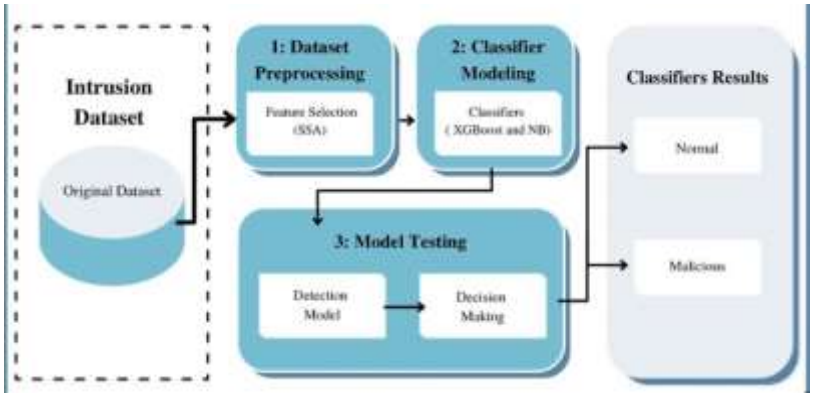

Figure 1

introduced techniques like the Alternating Direction Method of Multipliers (ADMM) to decompose large-scale problems into smaller, parallelizable subproblems. ADMM has emerged as a promising tool for anomaly detection due to its ability to handle sparse signal recovery, outlier separation, and regularized learning. Its robustness and scalability are suitable for network-level anomaly detection where interpretability, responsiveness, and precision are key. As traffic data becomes increasingly heterogeneous, structured, and encrypted, robust optimization tools are needed to separate meaningful anomalies from benign fluctuations.

2. LITERATURE SURVEY

Intrusion Detection Systems (IDSs) [1] are used for the protection of the network infrastructure. However, using traditional IDSs has several limitations in the IoT, as they were originally designed for traditional networks. Besides being resource-intensive, these methods, due to their static nature, cannot meet the requirements of the dynamic, heterogeneous, and resource-constrained nature of the IoT. Moreover, their limited ability to detect new types of attacks or low-frequency attacks makes them less useful in practice.

Evolution of Intrusion Detection Systems

The development of IDS technology dates back decades, starting with the earliest signature-based systems [2] that could match the characteristics of these cyberattacks to rules within the IDS. These systems operated very similarly to antivirus software, utilizing databases of known signatures to detect malware. While these systems are very effective in detecting familiar threats, they are essentially only capable of matching them against known threats and require frequent updates to detect new cyberattacks. For example, if a hacker is able to use malware (e.g., worms) or network attacks (e.g., base64 code bypass), it exposes the shortcomings of signature-based IDSs in identifying unknown attacks.

Anomaly-based detection [3] emerged in response to these challenges, aiming to identify normal network behavior when it deviates from an established baseline. Because the methodology introduces statistical models and heuristics, it is able to detect the ever-changing nature of network attacks. For example, techniques such as principal component analysis (PCA) [4] and clustering [5] algorithms are used to identify anomalies in network traffic patterns. Despite these advances, anomaly-based systems often face high false-positive rates. For example, network behavior during large-scale legitimate updates or migrations often triggers false alarms, highlighting the difficulty in defining a consistent baseline of “normal” behavior in a complex and volatile environment.

Subsequent hybrid systems have attempted to combine the strengths of feature-based and anomaly-based approaches to create multi-layer [6] approaches to improve detection accuracy. Moreover, in recent years, systems have begun to incorporate basic machine learning techniques, such as judgment

trees and support vector machines, to better analyze network traffic. While hybrid systems represent a significant improvement, their computational resources have increased, often limiting their scalability and applicability in resource-constrained environments.

AI-Based Intrusion Detection Systems

Numerous researchers have executed and integrated machine learning (ML) [7], deep learning (DL) [8], and AI methods in IDSs. AI-based IDSs can determine behavioral patterns. Intrusion Detection Systems are living systems and can be more effective than a simple statistical model at finding an intrusion. Experts first employed supervised learning techniques, such as Random Forests and Gradient Boosted Trees, to classify network data as either benign or malicious. These models were significantly more effective than the previous ones; they also produced fewer false alarms and required less computation time.

Deep learning has changed the picture significantly. Convolutional Neural Networks (CNNs) [9] and Recurrent Neural Networks (RNNs) [10] are used to look at temporal and special patterns in network traffic for attack detection. As opposed to looking for a specific pattern, using a temporal and special pattern allows these models to distinguish attackers from legitimate users. Long Short-Term Memory (LSTM) networks, a type of RNN, have been proven to be able to capture temporal dependencies in continuous data. Therefore, they can detect distributed denial-of-service attacks.

3. PROPOSED SYSTEM

The research work implements a robust machine learning pipeline for anomaly detection in IoT edge devices, leveraging Logistic Regression and a Decision Tree Classifier with AdaBoost. It begins by importing essential libraries for data processing, visualization, and modeling, followed by loading and exploring a CSV dataset to examine its structure and identify missing values. After preprocessing and visualizing class distribution, the data is split into training and testing sets using a 70-30 ratio.

A custom function is defined to calculate and visualize performance metrics, including accuracy, precision, recall, and F1-score. The research work trains or loads pre-saved models, evaluates them on the test set, and compares their performance in a summary table. Finally, it applies the trained AdaBoost-enhanced Decision Tree model to new test data for real-time predictions, demonstrating the system's applicability in practical IoT environments. This research presents a comprehensive machine learning pipeline for anomaly detection in IoT edge devices, utilizing Logistic Regression and a Decision Tree Classifier enhanced with AdaBoost. The process begins by importing essential libraries and loading a CSV dataset into a panda DataFrame, where basic statistics and missing values are examined to assess data quality. Data preprocessing involves selecting features and targets, followed by visualizing the class distribution to evaluate balance. The dataset is split into training and testing sets in a 70-30 ratio, and features are standardized to ensure optimal model performance.

For model implementation, the pipeline checks for existing pre-trained models—training and saving new ones if unavailable—for both Logistic Regression and AdaBoost-enhanced Decision Tree. Each model is evaluated on the test set using accuracy, precision, recall, and F1-score metrics, visualized through a custom function. A performance comparison table summarizes the results, highlighting the effectiveness of each model. Finally, the trained AdaBoost model is applied to unseen test data to generate real-time predictions, demonstrating the system's practicality and robustness in real-world IoT environments.

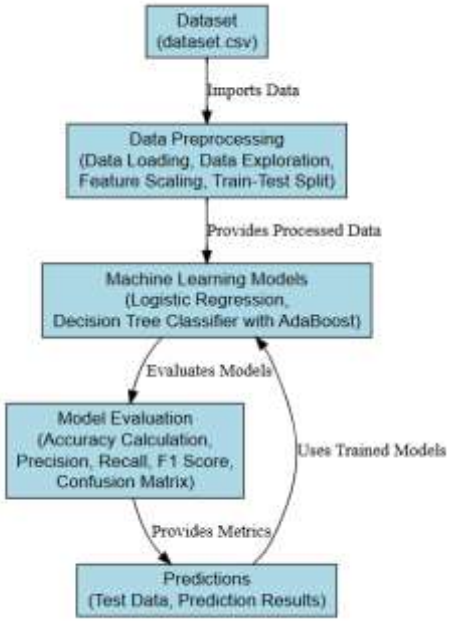

Fig.2: Architecture Diagram of Proposed method.

3.2 Data Preprocessing

Data preprocessing is a crucial phase in the machine learning pipeline for anomaly detection in IoT edge environments, ensuring the dataset is clean, well-structured, and suitable for model training. The process begins by separating the independent variables and the dependent variable ,which represent input features and the classification labels for anomaly types such as "Frequency Drift," "Capacity Breach," "Dual Signal Interference," and "Request Overload." A count plot is generated to visualize class distribution and detect any imbalances that might require corrective techniques like resampling or class weighting. Feature standardization is then applied using techniques like StandardScaler to normalize the input data, ensuring that features with varying units or scales do not disproportionately influence the model's learning process. This step is particularly vital for models like Logistic Regression that rely on gradient-based optimization. Finally, clearly defined class labels support interpretable evaluation through confusion matrices and classification reports, reinforcing the reliability and scalability of the anomaly detection system.

3.3 Proposed Decision Tree with AdaBoost Classifier

Decision Tree with AdaBoost Classifier is a powerful ensemble learning method that combines the strengths of decision trees and boosting algorithms. In this research, it is utilized to enhance the classification performance by focusing on difficult-to-classify instances. Decision Tree: The base classifier is a Decision Tree, which splits the data into subsets based on feature values to make predictions. Decision Trees can capture non-linear relationships and interactions between features. AdaBoost: AdaBoost (Adaptive Boosting) is a boosting algorithm that sequentially trains multiple weak classifiers (e.g., shallow Decision Trees) and combines them into a strong classifier. Each subsequent classifier focuses on the instances that were misclassified by previous classifiers. Weight Adjustment: AdaBoost adjusts the weights of misclassified instances, ensuring that the model pays more attention to hard-to-classify examples in subsequent iterations. Training and Evaluation: The model is trained using the AdaBoost algorithm with Decision Trees as base estimators. It is evaluated on the test set to assess its performance.

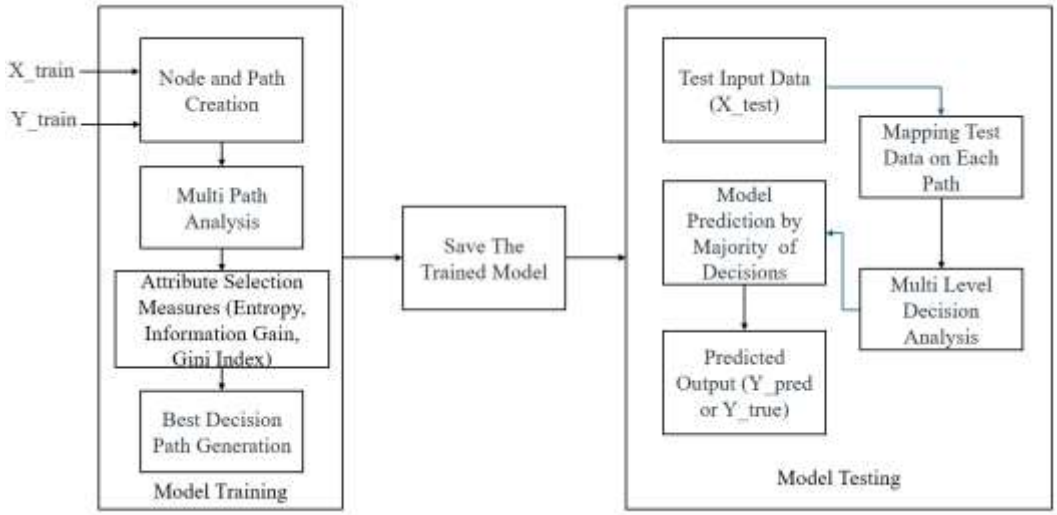

Fig.3: DTC Algorithm.

4. RESULTS AND DISCUSSION

This research implements a comprehensive Tkinter-based GUI application for anomaly detection in IoT network traffic, integrating data processing, machine learning, and interactive visualization. The application initializes a main window with labelled buttons that trigger core functions including dataset upload, preprocessing, data splitting, model training using Logistic Regression and AdaBoost with a Decision Tree base estimator, and real-time prediction. It utilizes global variables to manage state and maintain consistency across user actions. The preprocessing phase involves extracting features and the target label ("anomaly"), visualizing class distributions using Seaborn count plots, and applying standardization using StandardScaler. The dataset is split into training and testing sets using a 70-30 ratio, followed by model training, performance evaluation through classification reports and confusion matrix heatmaps, and saving/loading models using joblib for reuse. The AdaBoost model is employed for predicting anomalies on new data, displaying results in a scrollable text widget.

Figure 4 shows a count plot visualizing the distribution of the target labels in the dataset, generated in the preprocessing function using seaborn. count plot. The plot displays four categories (0 to 3) corresponding to the anomaly classes: "Frequency Drift," "Capacity Breach," "Dual Signal Interference," and "Request Overload.

The counts for each category are as follows: Category 0 ("Frequency Drift") has a count of 22140, Category 1 ("Capacity Breach") has 2584, Category 2 ("Dual Signal Interference") has 2465, and Category 3 ("Request Overload") has 2810. The plot uses a darkgrid style with a figure size of 8x6 inches, and the bars are coloured differently

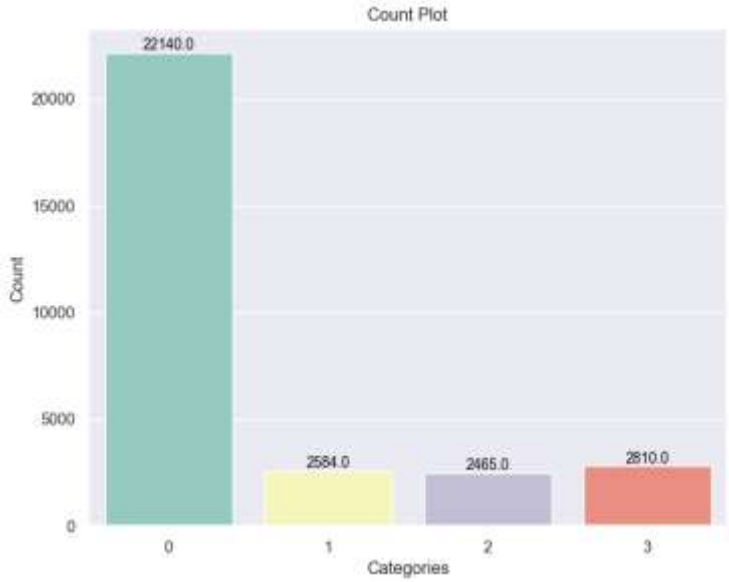

Fig.4: Count plot for the dataset.

Figure 5 shows the confusion matrix for the Decision Tree Classifier with AdaBoost (DTC with AdaBoost), generated in the AdaBoost function. The 4x4 matrix uses the same class ordering. The matrix values are: "Frequency Drift" (true) is predicted as 6642 (Frequency Drift), 0 (Capacity Breach), 0 (Dual Signal Interference), and 0 (Request Overload); "Capacity Breach" (true) as 0, 694, 0, 75; "Dual Signal Interference" (true) as 0, 0, 720, 0; "Request Overload" (true) as 0, 157, 0, 712. The heatmap, also in blue, shows near-perfect diagonal values (6642, 694, 720, 712), indicating high accuracy. Misclassifications are minimal, with 157 "Request Overload" instances predicted as "Capacity Breach" and 75 "Capacity Breach" instances predicted as "Request Overload." The predicted support values are 6642, 851, 720, and 787, aligning with the matrix sums. This figure highlights the AdaBoost model's superior performance compared to LRC.

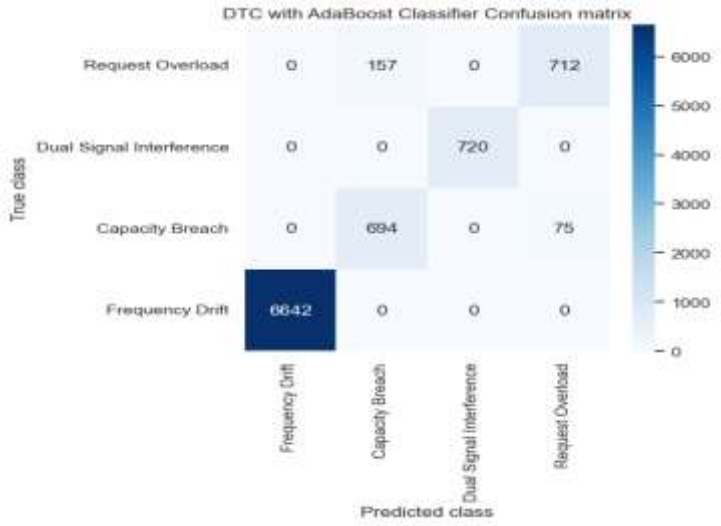

Fig.5: Proposed DTC with AdaBoost Confusion Matrix.

Table 1. Performance of Comparison of Various Classifiers.

Algorithm Name	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
Logistic Regression	40.71	50.12	43.30	68.58
DTC with AdaBoost Classifier	93.01	93.05	92.92	97.42

The comparison in table 1 presents a side-by-side evaluation of the Logistic Regression (LRC) and DTC with AdaBoost Classifier models based on their performance metrics: precision, recall, F1-score, and accuracy. The LRC model achieves an accuracy of 68.58%, with a precision of 40.71%, recall of 50.12%, and F1-score of 43.30%, reflecting moderate performance but significant struggles, particularly with "Request Overload" (0 correct predictions, as seen in Figure 9.4). In contrast, the DTC with AdaBoost Classifier excels with an accuracy of 97.42%, precision of 93.01%, recall of 93.05%, and F1-score of 92.92%. These metrics, supported by Figure 9.5's near-perfect confusion matrix, demonstrate the AdaBoost model's ability to handle all classes effectively, including "Request Overload" (712 correct predictions out of 869 true instances). The table, which would be generated by the compare function and displayed in the GUI's text widget, underscores the AdaBoost model's robustness in addressing class imbalances and complex patterns in network traffic data.

Table 2. Performance Analysis of Frequency Drift

Algorithm Name	Precision	Recall	F1-Score	Support
Existing LRC	0.79	0.85	0.82	6208
DTC with AdaBoost Classifier	1.00	1.00	1.00	6642

Table 3. Performance Analysis of Capacity Breach

Algorithm Name	Precision	Recall	F1-Score	Support
Existing LRC	0.62	0.25	0.35	1956
DTC with AdaBoost Classifier	0.90	0.82	0.86	851

Table 4. Performance Analysis of Dual Signal Interference

Algorithm Name	Precision	Recall	F1-Score	Support
Existing LRC	0.59	0.53	0.56	792
DTC with AdaBoost Classifier	1.00	1.00	1.00	720

Table 5. Performance Analysis of Request Overload

Algorithm Name	Precision	Recall	F1-Score	Support
Existing LRC	0.00	0.00	0.00	44
DTC with AdaBoost Classifier	0.82	0.90	0.86	787

For Frequency Drift, LRC achieves a precision of 0.79, recall of 0.85, and F1-score of 0.82 with a support of 6208, while DTC with AdaBoost achieves perfect scores (1.00 for all metrics) with a support of 6642, as seen in Figure 6 where all 6642 instances are correctly classified. For Capacity Breach, LRC struggles with a precision of 0.62, recall of 0.25, and F1-score of 0.35 (support 1956), with only 480 correct predictions whereas DTC with AdaBoost improves significantly with a precision of 0.90, recall of 0.82, and F1-score of 0.86 (support 851), correctly classifying 694 instances

For Dual Signal Interference, LRC has a precision of 0.59, recall of 0.53, and F1-score of 0.56 (support 792), with 423 correct predictions, while DTC with AdaBoost achieves perfect scores (1.00) with a support of 720, all correctly classified. For Request Overload, LRC fails completely (0.00 for all metrics, support 44), with no correct predictions (Figure 4), but DTC with AdaBoost performs well with a precision of 0.82, recall of 0.90, and F1-score of 0.86 (support 787), correctly classifying 712 instances (Figure 4). These tables highlight the AdaBoost model's superior performance across all classes, particularly in handling the challenging "Request Overload" class, which aligns with the class distribution observed.

CONCLUSION

The research work successfully demonstrates the application of machine learning techniques to identify anomalies in network traffic data, offering a user-friendly interface for data processing, model

training, and prediction. The tool leverages two classification models—Logistic Regression and Decision Tree Classifier with AdaBoost—to detect four types of anomalies: Frequency Drift, Capacity Breach, Dual Signal Interference, and Request Overload. The implementation using Tkinter provides an intuitive GUI that allows users to upload datasets, preprocess data, split it into training and testing sets, train models, and predict anomalies on new data. The Logistic Regression model achieves moderate performance with an accuracy of 68.58%, but struggles with certain classes due to class imbalance, particularly failing to classify Request Overload instances. In contrast, the DTC with AdaBoost Classifier significantly outperforms it, achieving an accuracy of 97.42% and demonstrating robust performance across all classes, even for the underrepresented Request Overload class. The tool's ability to save and load model weights ensures efficiency, while its detailed performance metrics (precision, recall, F1-score) and visualizations provide valuable insights into model behavior. Overall, this application effectively addresses the challenge of anomaly detection in network traffic, with the AdaBoost model proving to be a superior choice for handling complex and imbalanced datasets, making it a practical solution for network security monitoring.

REFERENCES

[1] Chen, T.M.; Blasco, J.; Alzubi, J.; Alzubi, O. Intrusion detection. Eng. Technol. Ref. 2014, 1, 1–9.

[2] Hubballi, N.; Suryanarayanan, V. False alarm minimization techniques in signature-based intrusion detection systems: A survey. Comput. Commun. 2014, 49, 1–17.

[3] Garcia-Teodoro, P.; Diaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges. Comput. Secur. 2009, 28, 18–28.

[4] Heba, F.E.; Darwish, A.; Hassanien, A.E.; Abraham, A. Principle components analysis and support vector machine based intrusion detection system. In Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications, Cairo, Egypt, 29 November–1 December 2010; pp. 363–367.

[5] Panda, M.; Patra, M.R. Some Clustering Algorithms to Enhance the Performance of the Network Intrusion Detection System. Estud. Econ. Appl. 2008, 26, 710–716.

[6] Awodele, O.; Idowu, S.; Anjorin, O.; Joshua, V.J. A Multi-Layered Approach to the Design of Intelligent Intrusion Detection and Prevention System (IIDPS). Issues Informing Sci. Inf. Technol. 2009, 6, 630–647.

[7] Wagh, S.K.; Pachghare, V.K.; Kolhe, S.R. Survey on intrusion detection system using machine learning techniques. Int. J. Comput. Appl. 2013, 78, 30–37.

[8] Ahmad, Z.; Shahid Khan, A.; Wai Shiang, C.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans. Emerg. Telecommun. Technol. 2021, 32, e4150.

[9] Mohammadpour, L.; Ling, T.C.; Liew, C.S.; Chong, C.Y. A convolutional neural network for network intrusion detection system. Proc. Asia Pac. Adv. Netw. 2018, 46, 50–55.

[10] Yin, C.; Zhu, Y.; Fei, J.; He, X. A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access 2017, 5, 21954–21961.