iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.84077

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

Advanced Network Traffic Analytics for Cyber Anomaly Detection Using Machine Learning

Dr. T. Chalama Reddy[1], K. Venkateswarlu[2]
[1]*Professor*, [2]*Assistant Professor, Department of Computer Science and Engineering, Geethanjali Institute of Science and Technology (Autonomous), Nellore, Andhra Pradesh, India*

Abstract: ***Network traffic anomaly detection plays an important role in improving cybersecurity by identifying unusual activities in network communication. This paper presents a machine learning-based system for detecting abnormal patterns in network traffic. The collected dataset is preprocessed to remove inconsistencies and prepare it for analysis, and Synthetic Minority Over-sampling Technique (SMOTE) is applied to balance the inherently imbalanced network traffic data and improve learning from minority attack samples. Both supervised and unsupervised machine learning algorithms, including Isolation Forest, Support Vector Machine, XGBoost, and LightGBM, are used to analyze network traffic and identify anomalies. The system is designed to detect suspicious activities such as cyber-attacks and abnormal traffic behaviour by learning patterns from network data, thereby improving the effectiveness and reliability of anomaly detection systems compared to traditional rule-based intrusion detection approaches.***

Keywords: Network Anomaly Detection, Machine Learning, Intrusion Detection System, SMOTE, Isolation Forest, XGBoost, LightGBM, Cybersecurity

I. INTRODUCTION

In today's digital world, network communication plays a vital role in connecting computers, servers, and devices across the Internet. With the rapid growth of digital technologies, cyber-attacks have also increased significantly. These attacks can disrupt network services, steal sensitive information, and cause financial and operational losses. Ensuring network security has therefore become a critical requirement for organizations and individuals.

Network traffic anomaly detection is an important technique used to identify unusual patterns in network communication. These unusual patterns may indicate malicious activities such as hacking, denial-of-service attacks, or unauthorized access. Traditional security systems rely on predefined rules and signatures to detect attacks; however, such systems often fail to detect new or unknown attack patterns. This work focuses on developing an advanced network traffic anomaly detection system using machine learning algorithms such as Isolation Forest, Support Vector Machine, XGBoost, and LightGBM.

Data preprocessing techniques such as SMOTE are used to handle imbalanced datasets and improve model performance. The goal is to detect cyber-attacks accurately and provide real-time predictions to enhance network security. The proposed system is designed to improve detection accuracy and reduce false alarms compared to traditional intrusion detection systems, and to support detection of both known and unknown cyber threats.

A. Motivation

The main motivation behind this work is the increasing number of cyber-attacks and security threats in modern computer networks. Organizations rely heavily on network communication for business operations, data storage, and online services, and any disruption in network services can lead to data loss, financial damage, and reputation loss. Traditional intrusion detection systems require manual configuration and cannot easily adapt to new attack patterns. As cyber threats continue to evolve, there is a need for intelligent systems that can automatically detect anomalies and respond quickly. Machine learning offers a powerful approach to this problem, since by analyzing historical network data, models can learn normal behaviour patterns and identify deviations from them.

The motivation for this work includes improving network security using intelligent techniques, detecting cyber-attacks automatically, reducing false alarms in intrusion detection systems, handling large volumes of network data efficiently, and supporting real-time monitoring and prediction.

B. Problem Statement

Network security has become a major concern due to the rapid increase in cyber-attacks. Traditional intrusion detection systems struggle to detect evolving cyber threats such as Denial of Service (DoS), Distributed Denial of Service (DDoS), port scanning, and malware attacks. These systems often produce a high number of false positives and fail to detect new or unknown attack patterns. Additionally, network datasets are usually imbalanced, with the number of normal traffic records considerably higher than the number of attack records. There is therefore a need for an advanced anomaly detection system capable of analyzing large volumes of network traffic data, handling imbalanced datasets effectively, detecting both known and unknown attacks, improving detection accuracy, and reducing false alarm rates.

C. Objectives

The main objectives of this work are to develop a machine learning-based system for detecting anomalies in network traffic; to implement both supervised and unsupervised learning algorithms for improved detection performance; to handle imbalanced datasets using SMOTE; to evaluate and compare the performance of different machine learning models; to achieve high detection accuracy with a low false alarm rate; to provide real-time predictions for network traffic classification; and to improve the overall security and reliability of computer networks.

II. LITERATURE SURVEY

Machine learning has become a widely used technology in cybersecurity, particularly for detecting anomalies in network traffic. Traditional intrusion detection systems rely on predefined rules and signatures, which are not capable of effectively detecting new or unknown threats. Research using publicly available datasets such as KDDCup'99 and NSL-KDD has applied algorithms including Naive Bayes, Support Vector Machine (SVM), Decision Tree, and Random Forest to demonstrate that machine learning can significantly improve intrusion detection performance by automatically learning patterns from network data.

Other studies have focused on the Isolation Forest algorithm, an unsupervised method designed specifically for anomaly detection that isolates unusual data points rather than relying on classification boundaries. Using features such as source and destination IP address, packet size, protocol type, and connection duration, this approach has been shown to identify suspicious activity automatically, highlighting the value of isolation-based methods in network security systems.

Feature selection has also been studied as an important step in anomaly detection pipelines, since network traffic datasets often contain a large number of attributes that do not contribute equally to prediction accuracy. Techniques such as correlation-based feature selection, Recursive Feature Elimination (RFE), Principal Component Analysis (PCA), and Information Gain have been used to reduce dimensionality, remove redundant features, and improve both processing speed and prediction accuracy.

Ensemble learning approaches using XGBoost and LightGBM have been explored for network anomaly detection, combining multiple models to improve prediction accuracy and reliability. Such studies, evaluated using accuracy, precision, recall, and F1-score, have shown that ensemble models achieve higher accuracy than traditional algorithms and are better able to detect complex patterns in network traffic data. Hybrid approaches combining supervised classification with unsupervised anomaly detection have also been investigated, with the goal of detecting both known and unknown cyber-attacks in real time. These hybrid systems, evaluated using accuracy, precision, recall, and F1-score, have demonstrated higher detection accuracy and reduced false alarm rates compared to single-model systems. The present work follows a similar hybrid approach, combining supervised algorithms (XGBoost and LightGBM) with an unsupervised algorithm (Isolation Forest) to detect different types of anomalies.

III. SYSTEM ANALYSIS

A. Existing System

Existing anomaly detection systems collect network traffic data from datasets such as KDDCup'99 and perform preprocessing steps including handling of missing values, encoding of categorical features, and feature scaling. Machine learning models such as Naive Bayes, SVM, Isolation Forest, XGBoost, and LightGBM are then trained to detect anomalies by analyzing traffic patterns and predicting whether traffic is normal or malicious based on learned patterns from historical data. Existing systems can detect only known attack patterns and are unable to effectively identify unknown or new cyber-attacks, and they generally do not address dataset imbalance. Limitations of existing systems include difficulty handling imbalanced datasets where attack data is limited, a high false positive rate, the need for high computational resources for complex models, reduced performance on high-dimensional traffic data, reliance on older datasets that may not represent modern threats, and longer training times with higher memory consumption.

B. Proposed System

The proposed system introduces an advanced machine learning-based model for detecting anomalies in network traffic that combines supervised algorithms (XGBoost, LightGBM) with an unsupervised algorithm (Isolation Forest) to detect different types of anomalies. The system is designed to detect both previously known attacks and new or unseen attack patterns. SMOTE is used to balance the dataset and improve model performance, while ensemble models such as XGBoost and LightGBM are used to achieve improved prediction performance compared to traditional models. The main improvement of the proposed system is the use of SMOTE to handle imbalanced datasets, which is intended to improve model accuracy and reduce bias.

The advantages of the proposed system include the ability to generalize to previously unseen attack types, efficient processing of high traffic volumes through LightGBM, robustness to imbalanced data that typically causes other models to fail, and reduced manual workload for security analysts through automated insights.

C. Functional and Non-Functional Requirements

The functional requirements of the system include data collection, data processing, model training and testing, model evaluation, and anomaly detection. The non-functional requirements include performance (processing large datasets quickly without delay), reliability (consistent operation and stable long-term performance), security (protection of sensitive network data and secure authentication), scalability (handling increasing data volumes), usability (a simple, user-friendly interface), availability (minimal downtime with backup and recovery), maintainability (modular design with clear documentation), and efficiency (effective use of memory and CPU resources).

D. Feasibility Study

1) *Economic Feasibility:* examines the project's expenses and feasible benefits to determine financial viability through cost-benefit analysis.
2) *Technical Feasibility:* evaluates the availability of resources, technology, and knowledge needed to deliver the project efficiently and on time.
3) *Social Feasibility:* evaluates the effect of the project on stakeholders and the community, considering social acceptance and broader impact.

IV. METHODOLOGY

The methodology describes the step-by-step process followed to develop the network traffic anomaly detection system, covering data collection, preprocessing, model training, evaluation, and prediction. Fig. 1 illustrates the overall system architecture.

A. Data Collection

The first step is collecting the dataset required for training and testing the machine learning models. The dataset contains network traffic records representing normal and abnormal network behaviour, stored in a structured CSV format that allows easy processing and analysis.

B. Data Preprocessing

Data preprocessing improves the quality of the dataset before applying machine learning algorithms by handling unnecessary or incorrect values. This includes removing missing values, eliminating duplicate records, normalizing data values, and selecting relevant features. SMOTE is applied after cleaning and preprocessing to balance the dataset by generating additional anomaly samples so that normal and anomalous classes are represented more equally, helping the models learn patterns from both classes effectively and reducing the risk of biased predictions.

C. Model Training and Evaluation

After preprocessing and balancing, the machine learning models are trained using the prepared data to learn patterns and relationships that distinguish normal from abnormal network traffic. The system uses Isolation Forest, XGBoost, and LightGBM. Once trained, model performance is evaluated using standard metrics, namely accuracy, precision, recall, and F1-score, and the model that provides the best performance is selected for anomaly detection.

D. Prediction and System Workflow

The best-performing model is used to predict whether new network traffic is normal or anomalous by comparing input data against patterns learned during training. Prediction results are displayed to the user through a web interface, allowing monitoring of network activity and identification of potential threats. The overall workflow proceeds as follows: the user uploads the network traffic dataset; the system preprocesses the data; machine learning models are trained using the dataset; the system evaluates model performance; the best model is selected; predictions are generated; and results are displayed to the user.

E. Algorithms Used

1) *Isolation Forest:* an unsupervised machine learning algorithm used for detecting anomalies by isolating abnormal data points from normal data patterns. It is efficient and well suited to large network traffic datasets.
2) *XGBoost:* a gradient boosting algorithm that improves prediction accuracy by combining multiple models and reducing errors. It is used to classify network traffic data and detect anomalies effectively.
3) *LightGBM:* a fast and efficient gradient boosting algorithm designed for large datasets, providing high performance and faster training speed compared to traditional methods, used to enhance the accuracy and speed of anomaly detection.

By using multiple machine learning models, the system improves prediction reliability and reduces the chances of incorrect classification.

V. SYSTEM REQUIREMENTS AND SPECIFICATIONS

A. Hardware Requirements

The hardware requirements include an Intel i5 processor, 8 GB of RAM, and 256 GB of hard disk storage.

B. Software Requirements

The software stack uses Python as the programming language, Flask as the frontend framework, Jupyter Notebook as the back-end development environment, SQLite3 as the database, and Windows as the operating system.

C. Packages and Libraries

The system makes use of Pandas and NumPy for data manipulation and numerical computation, Scikit-learn for machine learning algorithms and evaluation tools, TensorFlow and Keras for potential deep learning extensions, and Matplotlib for data visualization.

VI. SYSTEM DESIGN

Unified Modeling Language (UML) diagrams are used to represent the interaction between system components such as the User, Dataset, Application, Database, and System, helping to explain the workflow, data processing steps, and model execution.

A. Class Diagram

The class diagram represents the static structure of the system through classes including User, Dataset, Application, Database, and System. The User class handles registration, login, and input data; the Dataset class manages data used for training and testing; the Application class performs preprocessing, feature selection, model training, and prediction; the Database class stores credentials and processed data; and the System class generates the final classification output.

B. Sequence and Activity Diagrams

The sequence diagram describes the interaction between the User, Dataset, Application, Database, and System over time, beginning with importing packages and exploring the dataset, followed by preprocessing, visualization, label encoding, feature selection, train-test splitting, and model training using SVM, XGBoost, and LightGBM, before the system processes authenticated user input and returns the final prediction. The activity diagram represents the overall workflow from importing the dataset through preprocessing, visualization, label encoding, feature selection, train-test splitting, model training, user authentication, and generation of the final prediction indicating whether traffic is normal or anomalous.

C. Use Case Diagram

The use case diagram illustrates the interaction between the User and system functionalities, including registration and login, dataset selection or upload, data preprocessing, data analysis, model training, and prediction generation.

VII. IMPLEMENTATION

A. System Modules

The system is organized into the following modules: dataset collection, in which a network traffic dataset containing both normal and attack records (including DoS, DDoS, and port scanning types) is gathered for training and testing; data preprocessing, comprising data cleaning to remove duplicates and inconsistent records, feature selection and scaling to normalize inputs, and SMOTE-based handling of imbalanced data; model training, in which SVM, Isolation Forest, XGBoost, and LightGBM are trained using both supervised and unsupervised approaches, with ensemble models primarily used to achieve higher accuracy; model validation, in which the trained models are evaluated and compared using accuracy, precision, recall, and F1-score; anomaly detection, in which the final trained model classifies incoming traffic as normal or attack; and result generation and visualization, in which prediction results, attack probability, and threat level are presented, with SHAP values used to support interpretation of feature contributions.

B. Implementation Details

The implementation pipeline reads the network traffic dataset, encodes the binary traffic label, applies one-hot encoding to categorical fields such as protocol type, service, and flag, and performs Min-Max scaling on numeric features. Principal Component Analysis (PCA) is applied for dimensionality reduction while retaining 95% of variance, after which the training data is balanced by resampling the minority class to match the majority class. The processed and balanced data is then used to train the machine learning models, and a Streamlit-based web interface is used to load the trained model and processed data, allowing users to navigate between dataset information, preprocessing details, single-record analysis, and analytics pages to view predictions and explanations.

VIII. TESTING

Testing is a crucial process aimed at verifying that the system behaves as expected and meets specified requirements. It involves executing the software under controlled conditions to identify defects and validate that functionality, performance, and reliability requirements are satisfied. The testing objectives followed in this work are that testing is a process of executing a program to identify errors, that a good test case has a probability of finding an undiscovered error, and that a successful test uncovers an undiscovered error.

The testing principles applied include ensuring that all tests are traceable to end-user requirements, planning tests well in advance, progressing from small-scale to large-scale testing, recognizing that exhaustive testing is not possible, and, where feasible, having testing conducted by an independent party.

System testing, also referred to as black-box testing, focuses on the externally observable behaviour of the fully integrated system and validates it against end-user needs and requirements. This sits within the broader hierarchy of unit testing (testing a single software component), integration testing (testing a group of software units), system testing (testing the entire system), and acceptance testing (testing the acceptability of business requirements), with system testing occurring after integration testing and before acceptance testing.

IX. RESULTS

The system was implemented as an interactive web application that allows a user to explore the dataset, review the preprocessing pipeline, train and evaluate models, and analyze individual network packets. The dataset used in the implemented system corresponds to a KDD Cup 1999-style network intrusion detection benchmark, comprising a total of 132,072 samples split into a training set and a test set, with the original feature space comprising 42 raw attributes.

Feature categories examined include basic connection features such as duration, protocol type, service, flag, source and destination bytes; traffic features such as connection counts and error rates to the same host and service; content features such as failed login attempts, login status, and root-shell indicators; and host-based features such as destination host counts and service rates. The dataset includes multiple categories of attack records, broadly grouped as Denial of Service (DoS) attacks, Remote-to-Local (R2L) attacks, User-to-Root (U2R) attacks, and probing attacks, in addition to normal traffic records.

During preprocessing, the raw 42-feature representation was transformed through one-hot encoding of categorical fields and Min-Max scaling of numeric fields, followed by PCA-based dimensionality reduction that reduced the feature space to 29 principal components while retaining 95% of the variance. The training data was subsequently balanced so that normal and attack classes were represented more equally, addressing the class imbalance present in the raw dataset.

The trained model was evaluated on the held-out test set using accuracy, precision, recall, and F1-score as performance metrics. The implemented system reported an accuracy of 99.58%, precision of 99.76%, recall of 99.34%, and an F1-score of 99.55% on the test data, indicating that the model was able to distinguish effectively between normal and anomalous traffic in this evaluation setting.

The packet analysis interface allows a single test packet to be selected and analyzed, returning a classification (normal or attack) together with an attack probability and an associated threat level, alongside a recommended action. Feature contribution analysis using SHAP values was used to interpret individual predictions and to identify which principal components had the greatest influence on the model's decision for a given packet, supporting the explainability of the system's outputs.

These results indicate that the combination of SMOTE-based balancing, PCA-based dimensionality reduction, and ensemble or isolation-based learning algorithms is effective at distinguishing normal and anomalous network traffic on the evaluated dataset; further evaluation on additional or more recent network traffic datasets would help to confirm the generalizability of these results.

TABLE I MODEL PERFORMANCE METRICS ON THE TEST DATASET

Metric	Value
Accuracy	99.58%
Precision	99.76%
Recall	99.34%
F1-Score	99.55%

X. CONCLUSION

The proposed system detects network traffic anomalies using machine learning techniques to improve network security and identify potential cyber threats. By analyzing network traffic patterns, the system distinguishes between normal and abnormal activities, helping organizations monitor their networks and take preventive action. The system combines supervised and unsupervised machine learning models, and the use of ensemble learning techniques increases the reliability and stability of predictions by combining results from multiple models, reducing errors relative to a single-model approach.

Figure 1. shows the predicted output for a given sample

A key feature of the proposed system is the use of SMOTE to handle imbalanced datasets, since anomaly data is typically limited compared to normal traffic data; SMOTE helps balance the dataset by generating additional samples for the minority class, improving the learning ability of the models. The experimental results obtained on the implemented system demonstrate high detection accuracy and consistent precision, recall, and F1-score values, supporting the effectiveness of the approach. Overall, the system provides a reliable and efficient approach to improving network security and identifying cyber threats in a timely manner.

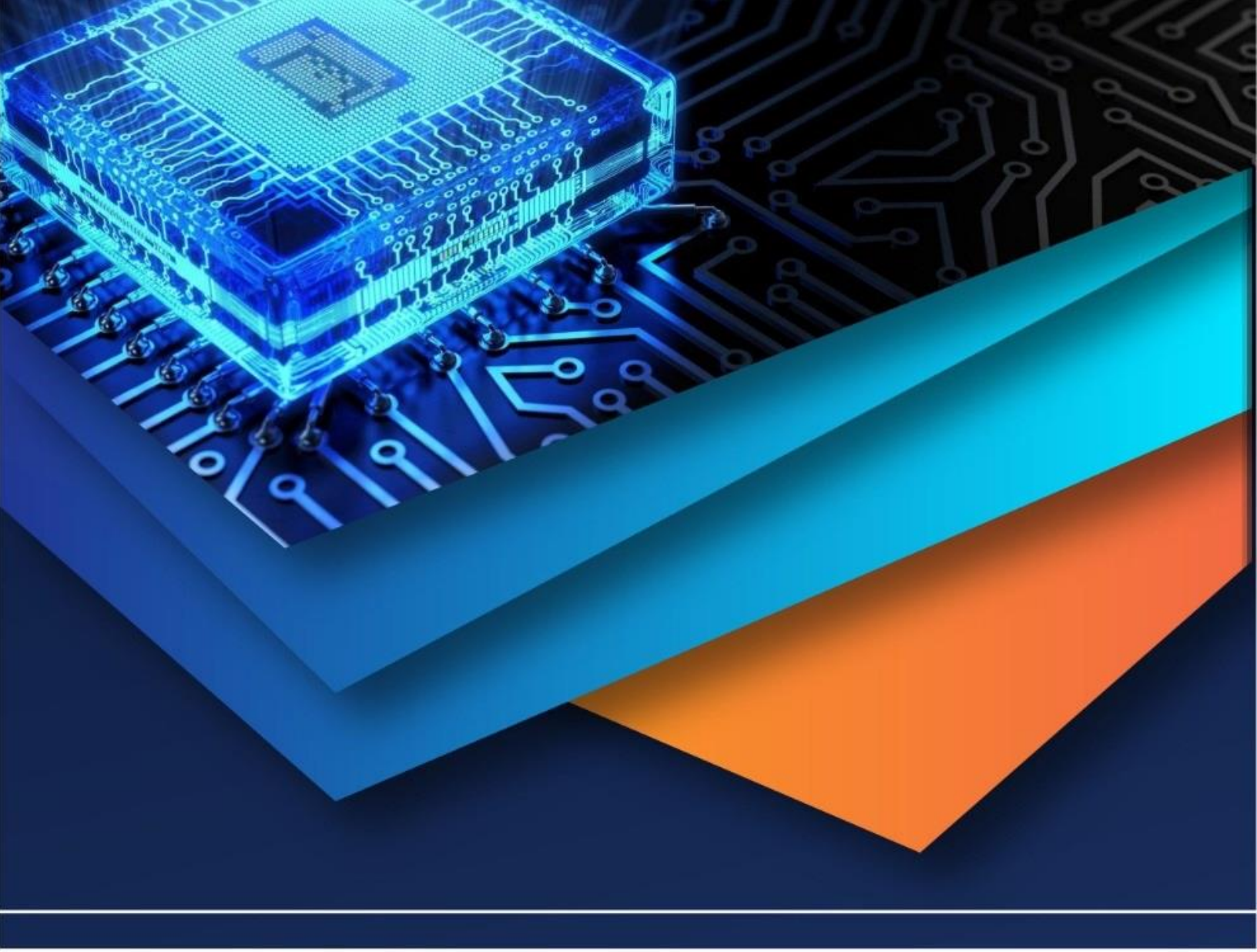
XI. FUTURE SCOPE

The system can be further improved by integrating deep learning models to enhance detection accuracy and to analyze complex network patterns and hidden relationships in large datasets, potentially improving the detection of sophisticated cyber threats. Implementation of real-time network monitoring would allow continuous analysis of network traffic and detection of anomalies as they occur, while deployment in cloud environments could improve scalability, storage capacity, and accessibility for large organizations and modern network infrastructures.

Future work may also focus on detecting new and advanced cyber-attacks more effectively by updating machine learning models regularly with new datasets, and on developing a user-friendly dashboard for network administrators that provides clear visualizations, alerts, and reports to support faster understanding of network conditions and timely security action.

REFERENCES

[1] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted Boltzmann machine," Neurocomputing, vol. 122, pp. 13–23, Dec. 2013.
[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection," ACM Comput. Surveys, vol. 41, no. 3, pp. 1–58, Jul. 2009.
[3] D. E. Difallah, P. Cudré-Mauroux, and S. A. McKenna, "Scalable anomaly detection for smart city infrastructure networks," IEEE Internet Comput., vol. 17, no. 6, pp. 39–47, Nov. 2013.
[4] E. Anceaume et al., "Anomaly characterization in large scale networks," in Proc. 44th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw., Jun. 2014, pp. 68–79.
[5] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," J. Netw. Comput. Appl., vol. 60, pp. 19–31, Jan. 2016.
[6] M. Usama et al., "Unsupervised machine learning for networking: Techniques, applications and research challenges," IEEE Access, vol. 7, pp. 65579–65615, 2019.
[7] K. Fotiadou et al., "Network traffic anomaly detection via deep learning," Information, vol. 12, no. 5, p. 215, May 2021.
[8] M. A. Umer, K. N. Junejo, M. T. Jilani, and A. P. Mathur, "Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations," Int. J. Crit. Infrastruct. Protection, vol. 38, Sep. 2022.
[9] A. A. Jihado and A. S. Girsang, "Hybrid deep learning network intrusion detection system based on convolutional neural network and bidirectional long short-term memory," J. Adv. Inf. Technol., vol. 15, no. 2, pp. 219–232, 2024.
[10] S. Naseer et al., "Enhanced network anomaly detection based on deep neural networks," IEEE Access, vol. 6, pp. 48231–48246, 2018.
[11] N. Kumar and S. Sharma, "A hybrid modified deep learning architecture for intrusion detection system with optimal feature selection," Electronics, vol. 12, no. 19, p. 4050, Sep. 2023.
[12]] S. Hajj et al., "Anomaly-based intrusion detection systems: The requirements, methods, measurements, and datasets," Trans. Emerg. Telecommun. Technol., vol. 32, no. 4, p. e4240, Apr. 2021.
[13] S. Gunupusala and S. C. Kaila, "Multi-class network anomaly detection using machine learning techniques," Contemp. Math., vol. 5, no. 2, pp. 5–22, Jun. 2024.
[14] V. Takale, A. Patil, A. Lonikar, A. Shinde, and P. V. Rupnar, "SecureNet: Network intrusion detection using machine learning and deep learning techniques," Int. J. Res. Appl. Sci. Eng. Technol., vol. 12, no. 3, pp. 439–443, Apr. 2024.
[15] S. H. Rafique, A. Abdallah, N. S. Musa, and T. Murugan, "Machine learning and deep learning techniques for Internet of Things network anomaly detection—Current research trends," Sensors, vol. 24, no. 6, p. 1968, Mar. 2024.
[16] M. Mynuddin et al., "Automatic network intrusion detection system using machine learning and deep learning," in IEEE MTT-S Int. Microw. Symp. Dig., Feb. 2024, pp. 1–9.
[17] S. Ness, V. Eswarakrishnan, H. Sridharan, V. Shinde, and N. V. P. Janapareddy, "Anomaly detection in network traffic using advanced machine learning techniques," IEEE Access, vol. 13, Dec. 2025.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY