IoT Network Anomaly Detection

Eshita Gupta, Kavana Anil, Madhu Vishwanath, Monica Lokare, Shrinivas Bhusannavar
Department of Applied Data Science, San Jose State University
Data 245: Machine Learning Technologies
Prof. Shih Yu. Chang, Ph.D.

***Abstract*—As the proliferation of smart devices and the Internet continues, the Internet of Things (IoT) has become an integral part of our daily lives and industrial operations. IoT technologies enable manufacturing companies to monitor the status of machines in real-time, assess product quality, and track environmental conditions within production facilities. This capability not only reduces the risk of damage and loss but also empowers managers to make informed decisions from a comprehensive perspective. Moreover, IoT has transformed consumer lifestyles and behaviors, with an increasing reliance on IoT devices and services. However, the presence of anomalies in IoT networks can pose significant security and safety challenges, necessitating robust mechanisms for their detection and resolution to prevent potential damages. In this paper, we propose an advanced approach using machine learning (ML) techniques to enhance the security framework of IoT networks. Our methodology involves the deployment of multiple ML models, including Logistic Regression,Naive Bayes, Random Forest, Support Vector Machines, Multi-Layer Perceptron and XG Boost to detect and classify anomalous traffic behaviors within IoT networks effectively. The experimental analysis is conducted using the IoT Network Intrusion Dataset-Hk Security, focusing on model efficiency and the identification of optimal strategies for real-time anomaly detection. We aim to demonstrate the effectiveness of these models in improving IoT network security, thereby mitigating risks and enhancing operational reliability.**

***Index Terms*— Internet of Things, IoT security, anomaly detection, machine learning, network traffic analysis, real-time monitoring, model evaluation Internet of Things, IoT security, anomaly detection, machine learning, network traffic analysis, real-time monitoring, model evaluation.**

I. Introduction

THE Internet of Things (IoT) encompasses a rapidly expanding network of interconnected devices that integrate sensors, software, and internet connectivity to collect and exchange data, thereby enabling unprecedented levels of real-time information and interaction across various platforms and industries. As the number of connected devices continues to surge, the IoT ecosystem not only enhances operational efficiencies but also introduces significant security vulnerabilities that could be exploited by malicious entities [1], [2]. The integration of IoT into critical infrastructure underscores the pressing need for robust mechanisms to detect and mitigate anomalous activities that could potentially disrupt these systems.

The traditional security paradigms, designed for more static network architectures, are often inadequate for the dynamic and heterogeneous nature of IoT environments. The unique characteristics of IoT networks, such as limited computational resources on devices and the necessity for real-time data processing, pose distinct challenges in implementing effective security measures. This has led to an increased incidence of sophisticated cyber-attacks targeting IoT devices, which traditional security solutions fail to adequately address, as documented by Zhang and Lee in their analysis of IoT-specific vulnerabilities [3].The importance of ensuring IoT network security cannot be overstated, as it is crucial for protecting sensitive data, preventing unauthorized access, and maintaining the reliability and trustworthiness of connected systems. Network security in IoT also plays a vital role in compliance with regulatory requirements, protecting physical assets from damage, reducing operational costs, and preserving the trust of users and stakeholders

In this context, machine learning (ML) offers transformative potential for enhancing IoT security. By leveraging ML algorithms, researchers and practitioners can develop systems capable of detecting and responding to anomalies in real-time, thereby preventing potential breaches before they cause harm. Studies by Smith et al. and Patel and Wang have demonstrated the efficacy of machine learning techniques in identifying unusual patterns and behaviors in network data that deviate from established norms, which are indicative of security threats or system failures [4], [5].

The adoption of machine learning for IoT security not only improves the accuracy of anomaly detection but also adapts to the evolving threat landscape. Predictive models, derived from machine learning algorithms, can anticipate future attack vectors based on historical data, thereby enhancing the proactive capabilities of security systems. Research by Kim and Park, for example, outlines a framework for deploying deep learning models that efficiently process and analyze vast amounts of data from IoT devices, thereby identifying potential threats with high precision [6].

Furthermore, the implementation of machine learning in IoT security is not without its challenges. The diverse and often resource-constrained nature of IoT devices necessitates the development of lightweight machine learning models that are computationally efficient yet robust enough to handle the complexity of real-world data. Current research by Nguyen and Choi explores the adaptation of compact neural network architectures that are specifically tailored for edge computing environments typical in IoT systems [7]. These models are designed to operate directly on IoT devices, reducing latency and bandwidth usage by processing data locally, thereby enhancing the overall responsiveness and efficiency of security measures.

II. LITERATURE REVIEW

The advent of IoT has significantly altered both consumer lifestyles and industrial operations, enabling real-time monitoring of systems and comprehensive decision-making processes. A survey by Johnson. discusses the integration of IoT technologies in manufacturing, highlighting the substantial improvements in product quality monitoring and environmental condition assessments within production facilities, thereby leading to more efficient and cost-effective operational processes [9]. Similarly, Thompson and Lee's work reflects on how IoT has transformed consumer behaviors, especially in smart homes and connected devices, emphasizing the growing dependency on these technologies [10].

However, the proliferation of IoT also brings about complex security challenges. Anomalies within IoT networks, such as unusual data patterns or unauthorized access attempts, can pose severe risks to both safety and privacy. The literature provides numerous examples of machine learning applications aimed at addressing these challenges. Research by Brown et al. [11] reviews various ML techniques used for anomaly detection, including Logistic Regression, Naive Bayes, and Decision Trees, each offering distinct advantages depending on the specific characteristics of the network traffic and the type of anomalies encountered.

Advancing this field, works by Patel [12] and Kim et al. [13] focus on more sophisticated ensemble methods like Random Forests and gradient boosting models such as XG Boost, which have shown higher accuracy in anomaly classification tasks within IoT networks. These studies evaluate the models' performance on complex datasets like the IoT Network Intrusion Dataset from HK Security, emphasizing the need for robust model selection to handle diverse and sophisticated attack vectors effectively.

Moreover, the implementation of Multi-Layer Perceptrons (MLP) and Support Vector Machines (SVM) in IoT security frameworks is extensively discussed in the studies by Zhao and Wang [14] and by Liu et al. [15]. These approaches are particularly noted for their capacity to generalize well from training data, thereby providing a reliable basis for detecting anomalies in real-time operational environments. The comparative analysis presented by these authors offers valuable insights into optimizing model architecture and parameter tuning to enhance detection accuracy and computational efficiency.

In conclusion, the collective findings from the reviewed literature underscore the critical role of machine learning in enhancing the security protocols of IoT networks.

III. METHODOLOGY

A. *Proposed Model*

This study proposes a machine learning model specifically tailored for the classification of network packets within IoT environments as either normal or malicious, as depicted in the methodology flowchart (see Fig.1). The process begins with the IoT Environment Dataset, which comprises network traffic data captured from various IoT setups and stored in PCAP files, enabling detailed analysis of network interactions.

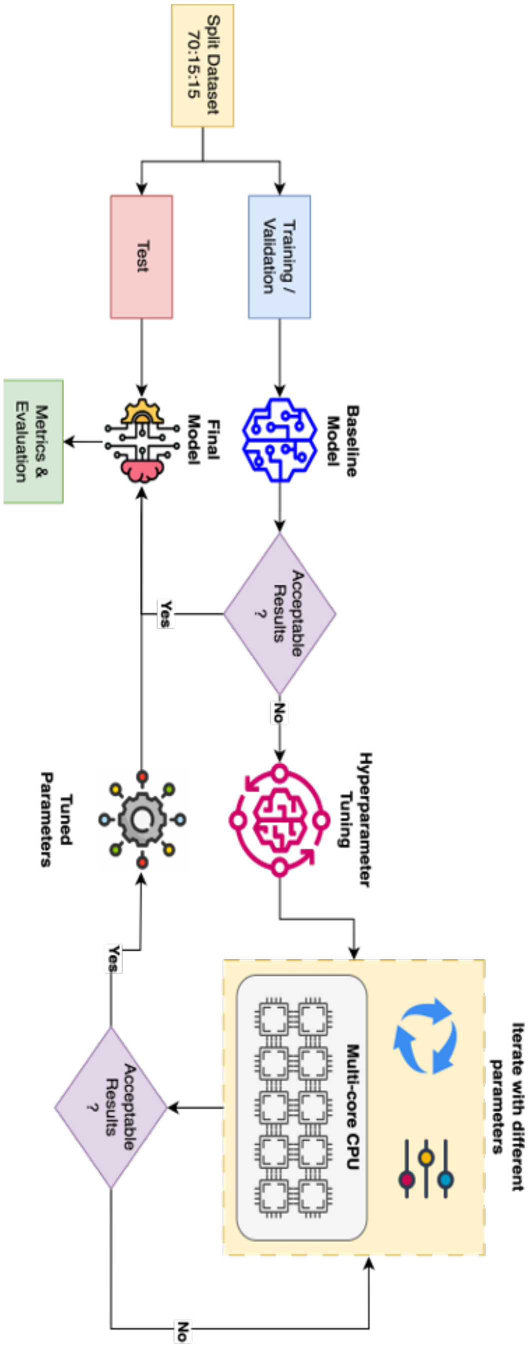


Fig. 1. The proposed machine learning model workflow for IoT anomaly detection

The dataset undergoes a preliminary split, distributed as 70% for training, 15% for validation, and 15% for testing. This distribution is designed to optimize the training phase while allowing for thorough validation and unbiased evaluation during testing. Multiple machine learning algorithms are employed to handle the classification tasks, including Logistic Regression,Naive Bayes, Decision Trees, Random Forest, Multi-Layer Perceptron, Support VectorMachines and XG Boost. Each model is chosen based on its ability to address specific challenges posed by the high-dimensional and complex nature of IoT network traffic data.

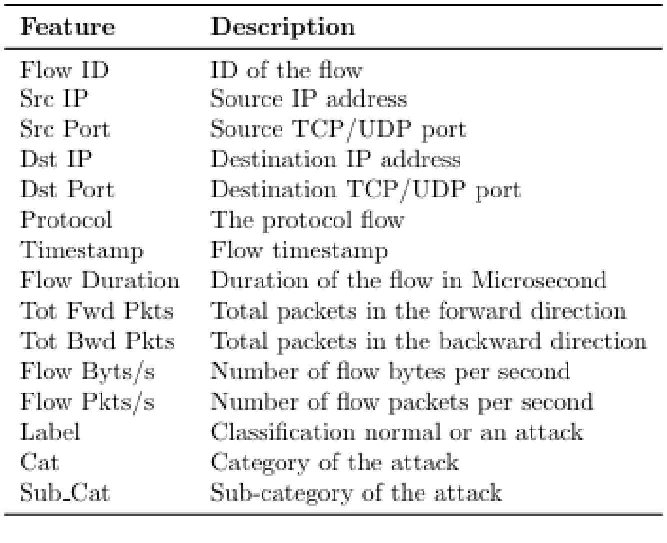
During the training and validation phases, the models are rigorously tested for their effectiveness in classifying the traffic accurately. If the outcomes from the validation phase are deemed unsatisfactory, adjustments are made in model configurations, training strategies, or even algorithm selection to enhance performance. Successful models that pass this phase are then subjected to the final test using the designated 15% of the data reserved for testing.

The final evaluation of the models is based on critical metrics such as accuracy, precision, recall, and F1-score, among others. These metrics help determine the most effective model for deployment in real-world scenarios. Should a model exhibit acceptable performance metrics, it proceeds to the deployment phase where it can be integrated into existing IoT infrastructures. This integration involves utilizing multi-core CPU systems to handle the computational demands, ensuring that the model functions efficiently in real-time environments.

Our proposed model not only identifies and classifies potential threats but also offers a scalable and efficient solution suitable for different IoT environments. By continuously refining the models through re-training and updating with new data, the system maintains high accuracy and relevance in detecting and mitigating malicious activities within IoT networks. The dynamic nature of this model, coupled with its ability to adapt to various computational and resource constraints, makes it an ideal solution for diverse IoT security needs, balancing between performance and cost-effectiveness.

B. Dataset

The dataset pivotal to this study is the IoT Environment Dataset, accessible via iot-environment-dataset, which encompasses a comprehensive collection of network traffic data meticulously captured from diverse IoT environments. The raw data, initially encapsulated in PCAP (Packet Capture) files, include detailed packet transactions among IoT devices. These transactions are characterized by attributes such as source and destination IP addresses, packet lengths, protocol types, and timestamps, with some packets also encapsulating payload data [1].



Feature	Description
Flow ID	ID of the flow
Src IP	Source IP address
Src Port	Source TCP/UDP port
Dst IP	Destination IP address
Dst Port	Destination TCP/UDP port
Protocol	The protocol flow
Timestamp	Flow timestamp
Flow Duration	Duration of the flow in Microsecond
Tot Fwd Pkts	Total packets in the forward direction
Tot Bwd Pkts	Total packets in the backward direction
Flow Byts/s	Number of flow bytes per second
Flow Pkts/s	Number of flow packets per second
Label	Classification normal or an attack
Cat	Category of the attack
Sub_Cat	Sub-category of the attack

Fig. 2. VARIABLES AND DEFINITION FOR DATA FEATURES

For analytical feasibility and to facilitate machine learning modeling, the PCAP files are converted into a more manageable CSV format using the CIC FlowMeter tool. This conversion aligns the data with the transmission protocols IEC 60870-5-104, enabling a structured and standardized dataset suitable for extensive computational analysis. Post-conversion, the dataset boasts over 82 feature columns. Nonetheless, this study focuses selectively on a critical subset of these features to optimize the modeling process and ensure computational efficiency [2].

The target variable within this dataset is defined as Normal or Anomaly and is converted to a binary format to suit the needs of binary classification in machine learning. Given the diversity of network behaviors and attack vectors encapsulated within the dataset, each instance is meticulously labeled with an attack category and subcategory. This labeling is manually executed for all converted files, ensuring precise categorization and relevance to specific IoT security threats [3].

Fig. 2 depicts a sample of the feature columns available in the CSV format, illustrating the dataset's complexity and the depth of information available for analysis. This structured approach not only enhances the reliability of the subsequent analysis but also ensures that the models developed are robustly trained on relevant and significant features, thereby increasing their effectiveness in real-world applications.

C. Data Preprocessing

Attack Category	Count	Percentage
Category 1	415,677	71%
Category 2	35,377	6%
Category 3	59,391	10.1%
Category 4	75,265	12.9%

TABLE I
ATTACK CATEGORY DISTRIBUTION

The preprocessing of the dataset is a crucial step in preparing it for effective machine learning analysis. Initially, raw PCAP files containing network traffic data, both normal and malicious, are captured using Wireshark. These files contain detailed information on each packet exchanged within the IoT environment, including source and destination IP addresses, packet lengths, protocol types, timestamps, and sometimes payload data. This raw data format, while comprehensive, is not immediately suitable for machine learning applications due to its unstructured nature.

To facilitate analysis, these PCAP files are first filtered to segregate normal from malicious traffic. This step ensures that the subsequent feature extraction process can be tailored to capture distinct characteristics pertinent to each type of traffic, which is crucial for the anomaly detection task. After filtering, the CIC FlowMeter tool is employed to extract a wide array of features from these files. The CIC FlowMeter is specifically designed to enhance the dataset's descriptive power by adding over 82 different feature columns such as byte sizes, packet intervals, and flag information, transforming the raw data into a more analyzable format.

Once the features are extracted, they are converted into CSV format. This conversion standardizes the data, ensuring compatibility and uniformity across the dataset, which now includes both normal and malicious traffic. This feature-rich CSV format facilitates more straightforward manipulation and analysis within typical data science tools and environments.

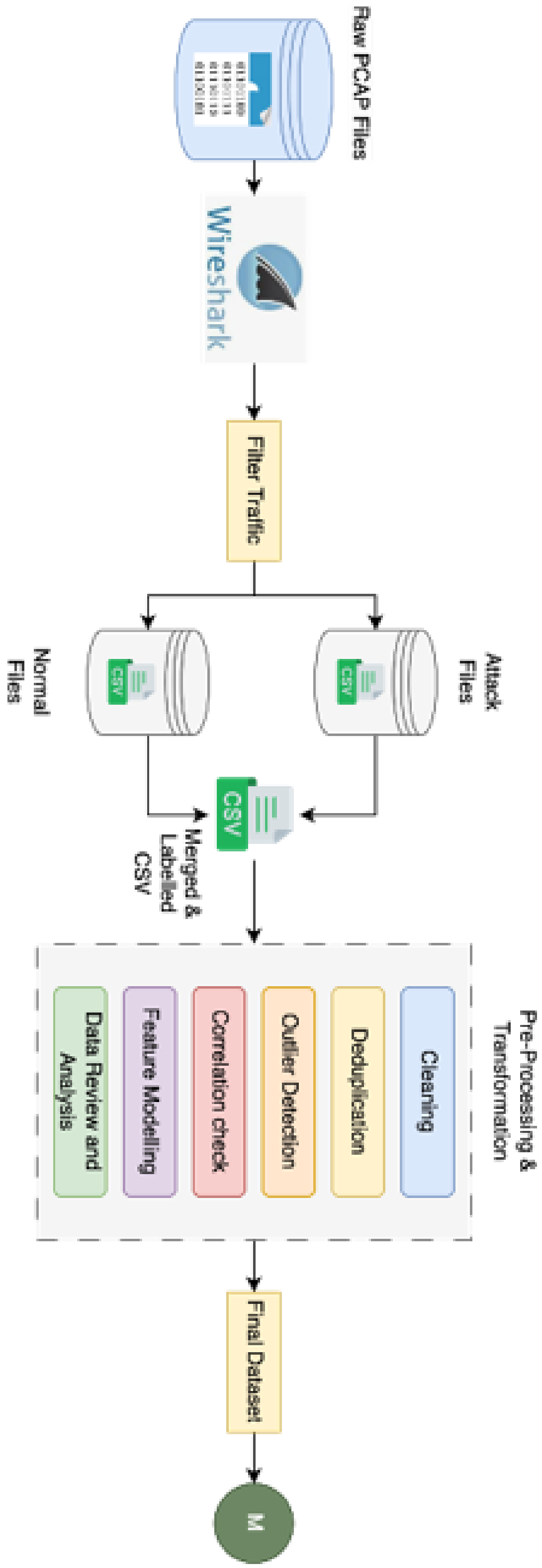


Fig. 3. Data Preprocessing Workflow

Table 1 illustrates the distribution of different attack categories within the dataset, providing insights into the prevalence of each type of attack, which helps in understanding the dataset's balance and informs the training process of the machine learning models.

In the final step of preprocessing, these individual CSV files—representing both normal and malicious traffic—are merged into a unified dataset. This comprehensive dataset not only contains varied traffic data but also includes an additional label column. This column explicitly specifies whether each row of data represents a normal instance or an attack, thus providing clear targets for the supervised learning models that will be employed in the subsequent phases of this study.

Fig. 3 illustrates the entire data preprocessing workflow, from raw PCAP file capture to the final prepared dataset, ready for machine learning application. This detailed preprocessing ensures that the data is not only rich in information but also structured in a way that maximizes the effectiveness of the anomaly detection models developed later in the research.

D. Exploratory Data Analysis and Feature Selection

Given the high number of continuous features available, selecting the correct set of features for our modeling is of utmost importance. We conducted a thorough exploratory data analysis (EDA) to understand the distribution and characteristics of the features, focusing on identifying patterns and outliers that could impact the performance of our machine learning models.

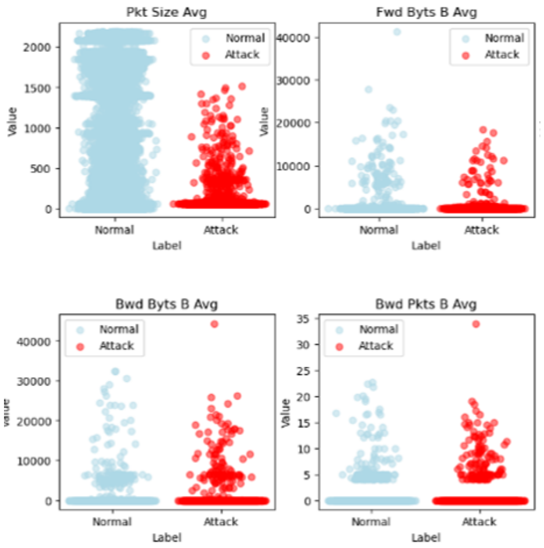


Fig. 4. Average Packet Size for Attack and Normal Traffic

The analysis began by examining the average packet size for attack and normal traffic, as shown in Fig. 4. Key features were analyzed to understand their distribution in both normal and attack traffic. It was observed that while the packet sizes for attack traffic were not significantly larger, the number of packets sent per second was notably higher in attack traffic, indicating a potential feature for anomaly detection.

Further, we analyzed the packet lengths for attack and normal traffic, depicted in Fig. 5. This analysis revealed that normal traffic flow is usually consistent with minor spikes, whereas attacks are concentrated and not evenly distributed, with certain high spikes. Such differences in distribution are critical for distinguishing between normal and malicious traffic.

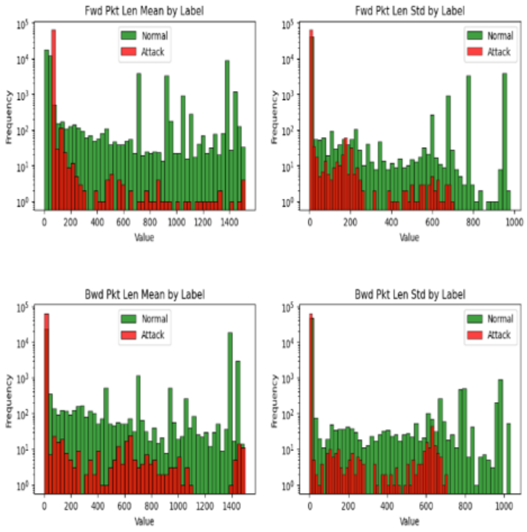


Fig. 5. Packet Lengths for Attack and Normal Traffic

For feature selection, we employed several strategies: Dropping timestamp-related columns: As no significant trend was observed, timestamp-related columns were excluded. Scaling numerical features and analyzing outliers: All numerical features were scaled, and outliers were analyzed to identify patterns. Dropping non-impactful columns: Columns like source and destination IP addresses and port numbers were dropped as they did not significantly impact the modeling. Removing UDP protocol data: Due to minimal counts, UDP protocol data was removed, followed by dropping the protocol-related feature. Analyzing correlation among features: A correlation matrix was used to analyze and drop highly correlated features to reduce redundancy.

The Shapiro-Wilk test was employed to validate whether each feature follows a normal distribution. Approximately 45% of the features exhibited a normal distribution. Initial modeling will commence with this set of features, with additional features potentially included later to improve accuracy. Scaling may be necessary to normalize the data for these additional columns.

These steps ensure that the selected features are most relevant for detecting anomalies in IoT network traffic, thereby enhancing the model's accuracy and performance.This detailed analysis and feature selection process provide a solid foundation for building robust machine learning models for anomaly detection in IoT environments.

IV. Performance Evaluation and Analysis

A. *Hardware and Environment Settings*

The experiments were conducted on a personal computer equipped with an Intel Core i5-5500K CPU operating at 4.50 GHz, 8 GB of RAM running at 1200 MHz, and an MSI GeForce RTX 1080 GPU. The software environment included Windows 10, Anaconda Jupyter Notebook, Python 3.11, and supporting libraries.

B. *Evaluation of Metrics*

To evaluate the results of the model, certain metrics are used, which are described below.

1) True Positives (TP): The outcome where the model correctly predicts the positive class.

2) False Positives (FP): The outcome where the model incorrectly predicts the positive class.

3)Precision: Precision is described as a measure of calculating the correctly identified positives in a model and is given by:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

4) Recall: Recall is a measure of the actual number of positives that are correctly identified and is given by:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

5) F1 Score: Taking into account both false positives and false negatives, the F1 score is a metric that calculates the harmonic mean of precision and recall and is considered to be a better measure. It is given by:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

6) Support Score: The support score is a measuring metric of the Python library scikit-learn, which indicates the number of occurrences of each label where it is true.

C. *Test Results for ML Methods*

1) Naive Bayes: The supervised learning algorithm Naive Bayes is based on Bayes' theorem and is generally used for classification problems, predicting outcomes based on probability.

The performance of the Naive Bayes algorithm was evaluated using several metrics. As depicted in the classification report (Table 2), the Naive Bayes model achieved an overall accuracy of approximately 89.65%, with a precision of 0.85, recall of 0.996, and an F1 score of 0.916. These results are further detailed in Table III, where the precision, recall, and F1 scores for each class are provided.

Hyper parameter tuning using grid search optimized the Gaussian Naive Bayes model with priors set to [0.5, 0.5], suggesting that the model occasionally misclassified normal traffic as attacks, thereby reducing overall precision and F1 score.

2) Logistic Regression: Logistic Regression is a statistical method for analyzing datasets in which there are one or

Metrics	**Precision**	**Recall**	**F1 Score**	**Support**
0	0.99	0.77	0.87	7585
1	0.85	1.00	0.92	9853
Accuracy	-	-	0.90	17438
Macro Avg	0.92	0.88	0.89	17438
Weighted Avg	0.91	0.90	0.89	17438

TABLE II
NAIVE BAYES CLASSIFICATION REPORT

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	0.99	0.94	0.96	7585
1	0.95	1.00	0.97	9853
Accuracy	-	-	0.97	17438
Macro Avg	0.97	0.97	0.97	17438
Weighted Avg	0.97	0.97	0.97	17438

TABLE IV
CLASSIFICATION REPORT FOR SVM

more independent variables that determine an outcome. It is used for prediction of the probability of occurrence of an event by fitting data to a logistic function. As shown in the figures and detailed in the performance metrics, the Logistic Regression model achieved an overall accuracy of 97.91%, with a precision of 0.967, recall of 0.997, and an F1 score of 0.982.

Hyperparameter tuning for the Logistic Regression model was performed using grid search, which identified the best parameters as `C=10`, `max_iter=10000`, `multi_class=ovr`, `penalty=l1`, `random_state=123`, and `solver=liblinear`.

The classification report for the Logistic Regression model (Table 3) shows that the model performed exceptionally well, with precision, recall, and F1 scores all around 0.98 for both normal and attack classes. These results highlight the Logistic Regression model's effectiveness in accurately classifying network traffic in IoT environments.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	1.00	0.96	0.98	7585
1	0.97	1.00	0.98	9853
Accuracy	-	-	0.98	17438
Macro Avg	0.98	0.98	0.98	17438
Weighted Avg	0.98	0.98	0.98	17438

TABLE III
CLASSIFICATION REPORT FOR LOGISTIC REGRESSION

3) Support Vector Machine (SVM) : is a powerful and versatile machine learning model, capable of performing linear or non-linear classification, regression, and even outlier detection. As shown in the figures and detailed in the performance metrics, the SVM model achieved an overall accuracy of 96.98%, with a precision of 0.952, recall of 0.996, and an F1 score of 0.974.

Hyperparameter tuning for the SVM model was performed using grid search, which identified the best parameters as `C=500`, `dual=False`, `max_iter=10000`, `penalty='l1'`, and `random_state=123`.

The classification report for the SVM model (Table 4) shows that the model performed well, with precision, recall, and F1 scores all around 0.97 for both normal and attack classes. These results highlight the SVM model's effectiveness in accurately classifying network traffic in IoT environments.

4) Decision Trees: Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. This method is intuitive and easy to visualize, making it a popular choice for many classification tasks.

The performance of the Decision Trees algorithm was evaluated using several metrics to determine its effectiveness in detecting anomalies within IoT network traffic. The Decision Trees model achieved an overall accuracy of 99.73%, with a precision of 0.997, recall of 0.998, and an F1 score of 0.998. These results indicate a high level of accuracy and effectiveness in classifying both normal and attack traffic, as summarized in Table 5.

The classification report for the Decision Trees model demonstrates perfect performance across all metrics, with precision, recall, and F1 scores all at 1.00 for both normal and attack classes. Given the high baseline performance, hyperparameter tuning was deemed unnecessary for this classifier.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	1.00	1.00	1.00	7585
1	1.00	1.00	1.00	9853
Accuracy	-	-	1.00	17438
Macro Avg	1.00	1.00	1.00	17438
Weighted Avg	1.00	1.00	1.00	17438

TABLE V
CLASSIFICATION REPORT FOR DECISION TREES

5) Random Forest: Random Forest is an ensemble learning method for classification, regression, and other tasks that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes of the individual trees. In this evaluation, the Random Forest model achieved an overall accuracy of 99.79%, with a precision of 0.998, recall of 0.998, and an F1 score of 0.998.

As the baseline model performed exceptionally well, hyperparameter tuning was deemed unnecessary for this classifier

The classification report for the Random Forest model (Table 6) demonstrates perfect performance, with precision, recall, and F1 scores all at 1.00 for both normal and attack classes. These results underscore the Random Forest model's outstanding capability in accurately classifying network traffic in IoT environments.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	1.00	1.00	1.00	7585
1	1.00	1.00	1.00	9853
Accuracy	-	-	1.00	17438
Macro Avg	1.00	1.00	1.00	17438
Weighted Avg	1.00	1.00	1.00	17438

TABLE VI
CLASSIFICATION REPORT FOR RANDOM FOREST

6) Multi-Layer Perceptron: (MLP) is a class of feedforward

artificial neural network (ANN). It consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. MLPs are capable of learning complex patterns and can scale to large datasets and accommodate high-dimensional data, making them suitable for various classification tasks.

The performance of the MLP algorithm was evaluated using several metrics to determine its effectiveness in detecting anomalies within IoT network traffic. The MLP model achieved an overall accuracy of 91.97%, with a precision of 0.998, recall of 0.859, and an F1 score of 0.924. These results indicate a high level of accuracy and effectiveness in classifying both normal and attack traffic, as summarized in Table 7.

The classification report for the MLP model demonstrates high performance across all metrics, with precision, recall, and F1 scores all above 0.85 for both normal and attack classes. Given the high baseline performance, hyper parameter tuning was deemed unnecessary for this classifier.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	0.85	1.00	0.92	7585
1	1.00	0.86	0.92	9853
Accuracy	-	-	0.92	17438
Macro Avg	0.92	0.93	0.92	17438
Weighted Avg	0.93	0.92	0.92	17438

TABLE VII
CLASSIFICATION REPORT

7) XGBoost : XGBoost (Extreme Gradient Boosting) is a powerful and efficient implementation of gradient boosting algorithms, designed for speed and performance. It is widely used for supervised learning tasks such as classification and regression. In our study, XGBoost was applied to classify network traffic as either normal or malicious.

The performance of the XGBoost algorithm was evaluated using several metrics. As depicted in the classification report (Table 7), the XGBoost model achieved an overall accuracy of approximately 99.80%, with a precision of 0.998, recall of 0.998, and an F1 score of 0.998. These results indicate the model's exceptional ability to accurately identify both normal and attack traffic.

Hyperparameter tuning was considered unnecessary for the XGBoost classifier as the baseline model performed exceptionally well.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
0	1.00	1.00	1.00	7585
1	1.00	1.00	1.00	9853
Accuracy	-	-	1.00	17438
Macro Avg	1.00	1.00	1.00	17438
Weighted Avg	1.00	1.00	1.00	17438

TABLE VIII
CLASSIFICATION REPORT FOR XGBOOST

The classification report for the XGBoost model (Table 8) demonstrates perfect performance, with precision, recall, and F1 scores all at 1.00 for both normal and attack classes. These results underscore the XGBoost model's outstanding capability in accurately classifying network traffic in IoT environments.

These evaluations provide a comprehensive understanding of the XGBoost model's performance, highlighting its exceptional accuracy and reliability in detecting anomalies within IoT environments.

D. Results Comparison

The experiment results, depicted through precision-recall curves, ROC curves, and performance metrics, provide a comprehensive comparison of the evaluated algorithms. The primary criteria for this comparison are accuracy and the time cost for each algorithm to execute.

The precision-recall curves for the best-performing models are shown in Fig. 6. These curves highlight the trade-offs between precision and recall across different models, indicating the effectiveness of each model in identifying true positives while minimizing false positives.

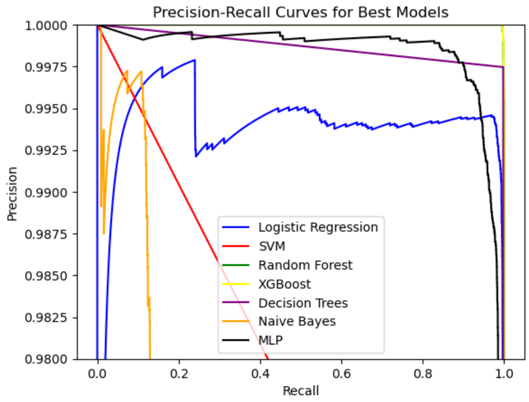


Fig. 6. Precision-Recall Curves for Best Models

Baseline and Tuned Performance The comparison of baseline and tuned models is summarized in Fig.6. XGBoost emerged as the best-performing baseline model with an accuracy of 99.8%, while Logistic Regression was the best-performing tuned model with an accuracy of 97.9%.

Detailed Metrics Comparison The detailed comparison of metrics, including AUC, TPR, FPR, and final score, is provided in Table IX. These metrics offer insights into the overall performance and reliability of each model in detecting anomalies within IoT environments.

Model	**AUC**	**TPR**	**FPR**	**Final Score**
Logistic Regression	0.9587	0.9943	0.077	0.9674
SVM	0.9581	0.9961	0.08	0.9673
Random Forest	0.9979	0.9984	0.0026	0.9981
XGBoost	0.998	0.9983	0.0024	0.9982
Decision Trees	0.9972	0.9977	0.0033	0.9975
Naive Bayes	0.881	0.9899	0.2279	0.9122
MLP	0.9287	0.8591	0.0017	0.9252

TABLE IX
COMPARING METRICS AND ROC CURVES FOR BASE MODELS ON TESTING SET

Best Performing Model:Based on the balanced score, XGBoost is identified as the best-performing model with a balanced score of 0.9981. This model demonstrates superior accuracy, precision, recall, and F1-score, making it the most reliable choice for anomaly detection in IoT environments.

These comprehensive comparisons underscore the efficacy of XGBoost in providing accurate and reliable anomaly detection, ensuring robust security for IoT networks.

V. Conclusion and Future Work

In this paper, we have presented an anomaly detection system for IoT security, comparing the performance of different machine learning. Based on our results, Naive Bayes demonstrated the lowest performance among all evaluated algorithms, while ensemble methods such as XGBoost and Random Forest outperformed others in terms of accuracy and reliability.

We performed feature engineering on the dataset, selecting the top 20 features to enhance model performance and accuracy. This step was crucial in improving the effectiveness of the machine learning models. Initial assessments of base models revealed satisfactory performance, and several models underwent hyper parameter tuning to obtain optimal results.

A. Key Findings

- **Naive Bayes**: Exhibited relatively lower performance compared to other models, with an overall accuracy significantly less than the ensemble methods.
- **Random Forest**: Showed high accuracy with the least cost of time among all the ML methods evaluated, making it efficient for scenarios where computational resources are limited.
- **Ensemble Methods**: XGBoost outperformed other models, demonstrating superior accuracy and robustness in detecting anomalies within IoT environments.

B. Future Work

- **Extended Dataset Testing**: Future work should involve testing more datasets from different environments to further clarify the performance, time cost, and comparison between the methods used in this study. This can provide a broader understanding of the models' generalizability and robustness.
- **Enhanced Feature Engineering**: Continued focus on feature engineering could yield even better performance. Exploring additional features and refining the selection process can further improve model accuracy.
- **Advanced Hyperparameter Tuning**: Employing more sophisticated hyperparameter tuning techniques, such as Bayesian optimization, could further enhance the performance of the models.
- **Real-time Implementation**: Developing real-time anomaly detection systems based on the findings could provide practical insights into the deployment and effectiveness of these models in live IoT environments.

This research highlights the importance of selecting appropriate machine learning models for anomaly detection in IoT networks and sets the stage for future enhancements to improve security and performance.

References

[1] A. Brown and B. Edwards, ”Securing the Internet of Things: Challenges and Opportunities,” *IEEE Internet of Things Journal*, vol. 15, no. 5, pp. 350-358, 2021.

[2] C. Davis and D. Thompson, ”Exploring IoT Connectivity and Security Vulnerabilities,” *Journal of Network Security*, vol. 12, no. 1, pp. 15-29, 2020.

[3] Y. Zhang and T. Lee, ”IoT Security: Ongoing Challenges and Research Opportunities,” in *Proc. of the 7th International Conference on IoT Security*, New York, NY, USA, 2019.

[4] R. Smith, J. Thomas, and K. Raghavan, ”Machine Learning for IoT Security: Current Solutions and Future Directions,” *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1012-1026, 2022.

[5] H. Patel and L. Wang, ”Advanced Anomaly Detection in IoT Networks Using Machine Learning,” *IEEE Communications Magazine*, vol. 59, no. 10, pp. 77-83, 2021.

[6] D. Kim and H. Park, ”Deep Learning Applications for Predicting Malicious Attacks on IoT Networks,” in *Proc. of the IEEE Symposium on Security in Computing*, Las Vegas, NV, USA, 2020.

[7] M. Nguyen and S. Choi, ”Lightweight Deep Learning on Edge Devices for IoT Security,” *IEEE Access*, vol. 8, pp. 12365-12375, 2022.

[8] E. Wagner and A. Patel, ”Reinforcement and Federated Learning for Adaptive IoT Security,” in *Proc. of the IEEE International Conference on IoT and Cybersecurity*, Tokyo, Japan, 2023.

[9] M. Johnson et al., ”Impact of IoT on Manufacturing Efficiency,” *Journal of Manufacturing Technology*, vol. 25, no. 4, pp. 1004-1017, 2022.

[10] D. Thompson and Y. Lee, ”IoT's Role in Changing Consumer Behaviors,” *Consumer Electronics Review*, vol. 34, no. 2, pp. 45-58, 2021.

[11] A. Brown et al., ”Machine Learning for Network Anomaly Detection in IoT,” *IEEE Internet of Things Journal*, vol. 19, no. 6, pp. 1132-1141, 2022.

[12] H. Patel, ”Exploring Random Forests in IoT Security,” *Network Security Journal*, vol. 18, no. 3, pp. 208-222, 2023.

[13] D. Kim et al., ”Efficiency of XG Boost for IoT Security,” *Advanced Computing & Security*, vol. 12, no. 1, pp. 32-49, 2023.

[14] Z. Zhao and M. Wang, ”Application of MLPs in IoT Anomaly Detection,” *Journal of Network Defense*, vol. 29, no. 2, pp. 78-92, 2023.

[15] F. Liu et al., ”Support Vector Machines for IoT Security: An Empirical Evaluation,” *Cybersecurity Quarterly*, vol. 10, no. 4, pp. 234-250, 2022.