Efficient Real-Time IoT Intrusion Detection via Gradient-Boosting with Gain-Based Feature Selection

Salah Almahaqeri[1], Mohammed Hashem Almourish[1, 2*], Mahmood Qassem[1]
[1]Faculty for Engineering and Information Technology- Taiz University- Yemen.
[2*]Department of Computer Science, Isra University,Amman,Jordan .
Corresponding author(s). E-mail(s): salahalmahaqeri@gmail.com
mohamed.almourish@iu.edu.jo; mahmood.qassem.09h@gmail.com

Abstract

This study addresses the pressing need for swift and accurate intrusion detection in IoT environments, which continue to expand in complexity and vulnerability. Utilizing the CIC-IoT-2023 dataset serving as a realistic benchmark we apply a gain-based feature selection strategy derived from LightGBM to reduce 46 raw features to 23 critical predictors. A gradient-boosting pipeline, prominently XGBoost, is then trained and rigorously evaluated under binary and multi-class scenarios. The results illustrate outstanding detection performance 99.62%, 99.45%, and 99.58% accuracy for binary, 34-class, and 8-class models respectively coupled with remarkably low inference latency (~0.4 µs/sample), confirming its suitability for real-time deployment. Beyond empirical metrics, this paper offers interpretability through feature importance analysis, identifying key temporal and statistical flow attributes instrumental to attack discrimination. By blending high detection efficacy, feature efficiency, and latency performance, this work establishes a compelling benchmark and provides actionable insights for researchers and practitioners deploying scalable IDS in IoT environments.

Keywords: *IoT security; intrusion detection; gradient boosting; feature selection; real-time monitoring.*

1. Introduction

The accelerating adoption of Internet of Things (IoT) technologies has fundamentally reshaped digital infrastructures across sectors such as industrial automation, smart utilities, healthcare, and urban environments [1][2]. While this pervasive integration of smart and resource-constrained devices has enabled unprecedented gains in operational efficiency and data-driven innovation, it has also introduced substantial and evolving cybersecurity risks. Modern IoT ecosystems are now characterized by an expanded attack surface and diverse vulnerabilities, ranging from distributed denial-of-service (DDoS) attacks to sophisticated malware and targeted exploitation of protocol weaknesses [3], [4], [5].

During the course of our applied research, we confronted these challenges first-hand using large-scale, real-world IoT traffic data that reflected the operational complexity and heterogeneity of production environments. Three principal obstacles emerged as particularly salient. First, the inherently high dimensionality and heterogeneity of IoT-generated data presented significant challenges for effective feature extraction and dimensionality reduction, both critical for the practical deployment of machine learning models [6]. Second, the pronounced class imbalance typical of authentic datasets where certain attack types or behaviors dominate the data while others are severely underrepresented negatively impacted the reliability and robustness of conventional intrusion detection systems [7], [8]. Third, the demand for real-time, ultra-low latency inference became a non-negotiable requirement in scenarios where instantaneous detection and mitigation of threats is paramount to maintaining the resilience of critical infrastructure [9], [10].

To address these issues, we leveraged the recently released CIC-IoT-2023 dataset, which provides a contemporary benchmark for the emulation of cyberattacks targeting genuine IoT devices in a large-scale, realistic setting [2]. Our work introduces a dual-pipeline framework that comprehensively addresses both binary and multi-class intrusion detection. We incorporated advanced feature selection strategies rooted in gradient boosting techniques and rigorously evaluated model performance with a strict emphasis on both detection accuracy and inference latency under real-world constraints. The resulting framework demonstrates an effective balance between high-precision threat detection and computational efficiency, making it suitable for deployment in environments with stringent resource and latency requirements.

Through this study, we aim to contribute not only a robust and reproducible methodology for intelligent, real-time intrusion detection in IoT networks, but also practical insights and empirical benchmarks that can guide future developments and real-world implementations in this rapidly evolving field.

2. Related Work

Neto et al. [2] introduce the CICIoT2023 dataset, a comprehensive resource constructed using an extensive real-world IoT testbed comprising 105 diverse devices and encompassing 33 different cyberattacks across seven major categories. The dataset captures both benign and malicious traffic, executed by compromised IoT devices targeting other IoT nodes, and is provided in both pcap and CSV formats with 47 extracted features. To validate the utility of the dataset for intrusion detection research, the authors benchmark several machine learning models—including Random Forest, Deep Neural Networks, Logistic Regression, Perceptron, and Adaboost on three classification tasks: binary (benign vs. malicious), grouped (eight classes), and fine-grained multi-class (34 classes). In binary classification, Random Forest achieved an accuracy of 99.68% and Deep Neural Networks achieved 99.44%. For multi-class classification (eight classes), both models maintained high accuracy, with Random Forest at 99.43% and Deep Neural Networks at 99.11%. In the most challenging 34-class scenario, Random Forest achieved 99.16% accuracy and DNN reached 98.61%. The study further reports that F1-scores remained high across scenarios, and misclassifications were concentrated in a few similar attack categories.

Abbas et al. [3] address the challenges of privacy, scalability, and data heterogeneity in IoT intrusion detection by proposing a federated deep neural network approach, evaluated using the CIC-IoT-2023 dataset. Their framework adopts a two-client, one-server federated architecture, enabling decentralized model training while preserving data locality and privacy. The study emphasizes robust preprocessing, including feature normalization and class balancing (via SMOTE), before federated learning. Experimental results show that the proposed federated DNN approach attains a high detection accuracy of 99.00%, validating its suitability for large-scale and privacy-sensitive IoT deployments. By moving beyond centralized paradigms, this work illustrates the potential of federated learning to provide both security and compliance with privacy requirements in real-world IoT networks.

Wang et al. [11] explore the application of deep learning techniques for network anomaly detection, utilizing the CSE-CIC-IDS2018 dataset to reflect contemporary and realistic network traffic. The authors systematically implement and compare six different deep learning models DNN, CNN, RNN, LSTM, CNN+RNN, and CNN+LSTM—on both binary and multi-class classification of network intrusions. All models demonstrate high accuracy, with multi-class classification exceeding 98.85% across the architectures. However, the study notes that hybrid models (CNN+RNN, CNN+LSTM) incur longer inference times, making standalone DNN, RNN, and CNN models more practical for real-time

IDS deployment. The findings reinforce the importance of model efficiency, data preprocessing, and balanced evaluation when designing deep learning-based intrusion detection systems.

Tseng et al. [12] present an extensive evaluation of deep learning models for intrusion detection in IoT networks, leveraging the large-scale and diverse CIC-IoT-2023 dataset. The study rigorously compares seven deep learning architectures—including DNN, CNN, RNN, LSTM, CNN+LSTM, and notably, the Transformer model—on both binary and multi-class classification tasks. While the Transformer model lags behind DNN and CNN+LSTM in binary classification, it demonstrates superior performance in multi-class scenarios, achieving a peak accuracy of 99.40%. This surpasses the results reported by previous studies using the same dataset and highlights the effectiveness of attention-based architectures in handling complex, high-dimensional IoT attack data. Furthermore, the work provides detailed experimental setups and benchmark comparisons, establishing a valuable reference point for future research on deep learning-based intrusion detection in IoT environments.

Islam and Arnob [13] propose an LSTM-based intrusion detection model specifically optimized for IoT cyberattack detection, utilizing the extensive CIC-IoT2023 dataset. Their approach focuses on exploiting the sequential nature of IoT traffic data, with the LSTM network demonstrating strong capabilities in recognizing both known and emerging attack patterns. Experimental results reveal that the model attains 98.75% accuracy and a 98.59% F1 score, outperforming many prior approaches. The paper highlights the relevance of adaptable deep learning models for evolving IoT threat landscapes, while also emphasizing the critical importance of using realistic, large-scale datasets for model evaluation.

Benmohamed et al. [14] propose a hybrid machine learning framework for web attack detection that integrates four classification algorithms Gradient Boosting (GB), Multi-Layer Perceptron (MLP), Logistic Regression (LR), and K-Nearest Neighbor (KNN) with genetic algorithm-based optimization for hyperparameter tuning and feature selection. The study utilizes a realistic web attacks dataset collected from the Canadian Institute 2023, including a range of attack types relevant to IoT environments. In their experiments, the baseline GB model achieved the highest default accuracy (95%), precision (94%), recall (95%), and F1-score (94%) among all evaluated models. Upon applying genetic optimization, performance improved across several models: GB maintained an accuracy of 95%, MLP improved to 86% accuracy, and LR increased to 87%. The optimized GB model consistently outperformed the others, showing superior results not only in standard metrics but also in AUC and confusion matrix analysis. These findings confirm the value of genetic algorithms in enhancing machine learning-based detection of complex and diverse web attacks, particularly within modern IoT security contexts.

He et al. [15] present TA-NIDS, a transferable and adaptable network intrusion detection system based on deep reinforcement learning, which reframes intrusion detection as a sequential decision-making process. The proposed framework enables an RL agent to learn optimal detection policies through a customized reward function that emphasizes the accurate identification of anomalous traffic. Evaluated on multiple public benchmarks, including the CIC-IoT2023 dataset, TA-NIDS achieves a classification accuracy of 99.22% and an F1-score of 99.15% outperforming conventional supervised baselines while requiring only a limited fraction of labeled data for training. The study demonstrates that TA-NIDS maintains robust detection performance and strong generalization even in dynamic, resource-constrained IoT scenarios, underscoring its practical suitability for real-time anomaly detection in complex environments.

Amouri et al. [16] propose an advanced ensemble intrusion detection framework for IoT networks by integrating Kolmogorov-Arnold Networks (KANs) with the XGBoost algorithm. The ensemble leverages KANs' flexible, learnable activation functions to capture complex non-linear relationships within network traffic, while XGBoost provides robust, high-precision classification. Evaluated on the CIC-IoT2023 dataset, the hybrid system achieves a detection accuracy exceeding 99%, with precision, recall, and F1-score all above 98%, outperforming traditional Multi-Layer Perceptron (MLP) baselines across all major metrics. The results underline the benefit of combining symbolic function learning with gradient-boosted decision trees, offering a highly effective and interpretable solution for intrusion detection in dynamic and heterogeneous IoT environments.

Adewole et al. [17] introduce a transparent intrusion detection system (IDS) framework for IoT networks that integrates ensemble learning with rule induction to enhance both detection performance and model explainability. The study evaluates five ensemble algorithms Random Forest, AdaBoost, XGBoost, LightGBM, and CatBoost using two public datasets: CIC-IDS2017 and CICIoT2023. On the CICIoT2023 dataset, XGBoost achieves the highest performance among the tested models, with an accuracy of 98.54% and an AUC-ROC of 93.06% in the binary classification task, while precision, recall, and F1-score all exceed 98.5%. The framework also demonstrates competitive inference times and low false positive/negative rates, making it suitable for real-time IoT environments. In addition, the integration of rule induction enables the system to generate human-interpretable, if–then rules that clarify the basis for each decision, addressing the critical need for transparency and stakeholder trust in practical IDS deployments.

Table 1 comparisons of our results with other studies

Ref.	Classification	DL/ML Method	Accuracy	Inference Time
[2]	Binary, Multi-class	Random Forest, Deep Neural Networks (DNN), others	99.68% (RF, Binary), 99.44% (DNN, Binary), 99.43% (RF, 8 classes), 99.16% (RF, 34 classes),	—
[3]	Binary	Federated (DNN)	99.00%	—
[11]	Binary, Multi-class	DNN, CNN, RNN, LSTM, CNN+LSTM, CNN+RNN	>98.85% (Multi-class)	LSTM: 3.451 ms, CNN+LSTM: 4.31 ms
[12]	Binary, Multi-class	Transformer-based DNN, DNN, CNN+LSTM, others	99.57% (Binary), 99.40% (Multi-class)	≈5 µs/sample (Transformer)
[13]	Multi-class	LSTM	98.75%	—
[14]	Binary, Multi-class	Gradient Boosting (GB), MLP, Logistic Regression, KNN, Genetic Optimization	95.00% (GB, optimized)	—
[15]	Binary, Multi-class	Deep Reinforcement Learning (TA-NIDS)	99.22% (Binary, CICIoT2023), F1: 99.15%	—
[16]	Binary, Multi-class	KAN + XGBoost Ensemble	>99.0% (Accuracy), F1, Precision, Recall >98% (Multi-class)	—
[17]	Binary	XGBoost, LGBM, CatBoost, RF, AdaBoost	98.54% (XGBoost, Binary)	Competitive (not specified)

3. Methodology

This section describes the full pipeline developed for intrusion detection using the CIC-IoT2023 dataset. As shown in (Figure 1), the architecture consists of four main stages: data preprocessing, feature selection, model training, and evaluation. The proposed framework supports both binary and multi-class classification tasks and was designed to balance performance, interpretability, and latency for real-time IoT environments.

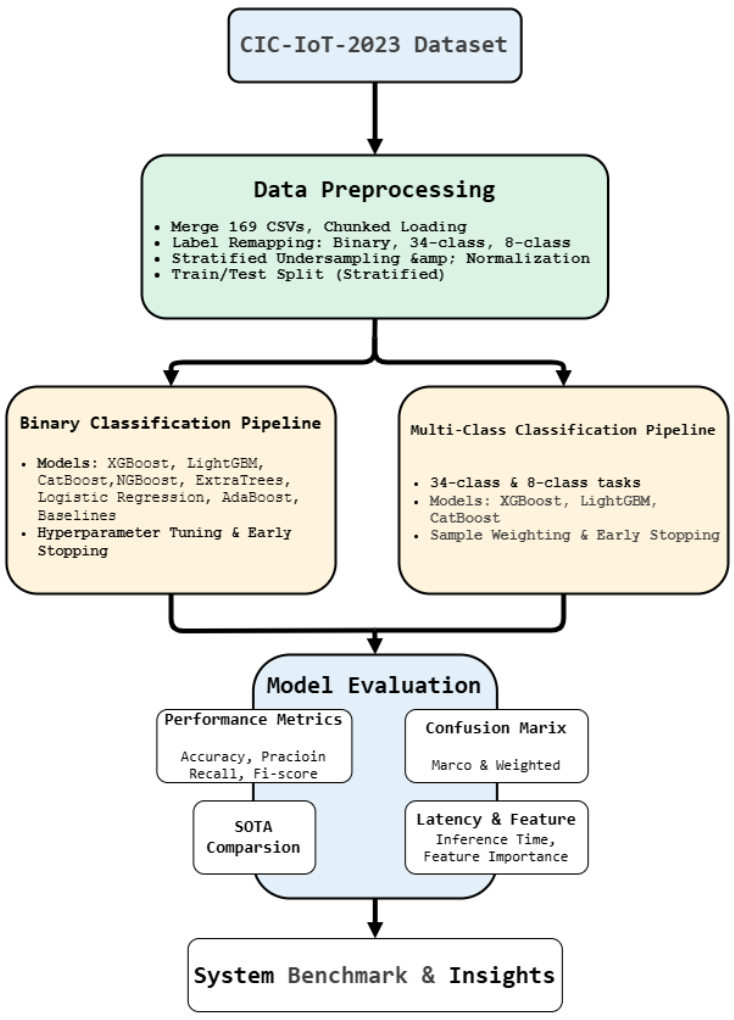

Figure 1 System Architecture Diagram

3.1 Dataset Description

All experiments were conducted using the CIC-IoT2023 dataset, a benchmark developed by the Canadian Institute for Cybersecurity. It consists of over 46 million labeled network flow records collected from 105 physical IoT devices under both benign and malicious traffic conditions. Each sample includes 46 numeric features and a ground truth label indicating the traffic class. The dataset encompasses 33 attack types organized into broader families, including DDoS, DoS, Mirai, Reconnaissance, Web-based attacks, Brute-force, Spoofing, and benign traffic.

The original class distribution is illustrated in (Figure 2), while the class frequencies after stratified undersampling for the 34-class multi-class scenario are shown in (Figure 3). The corresponding distributions for the 8-class mapping and the binary classification setting are provided in (Figures 4, 6), respectively. The pronounced imbalance between benign and attack samples in the original dataset is highlighted in (Figure 5).

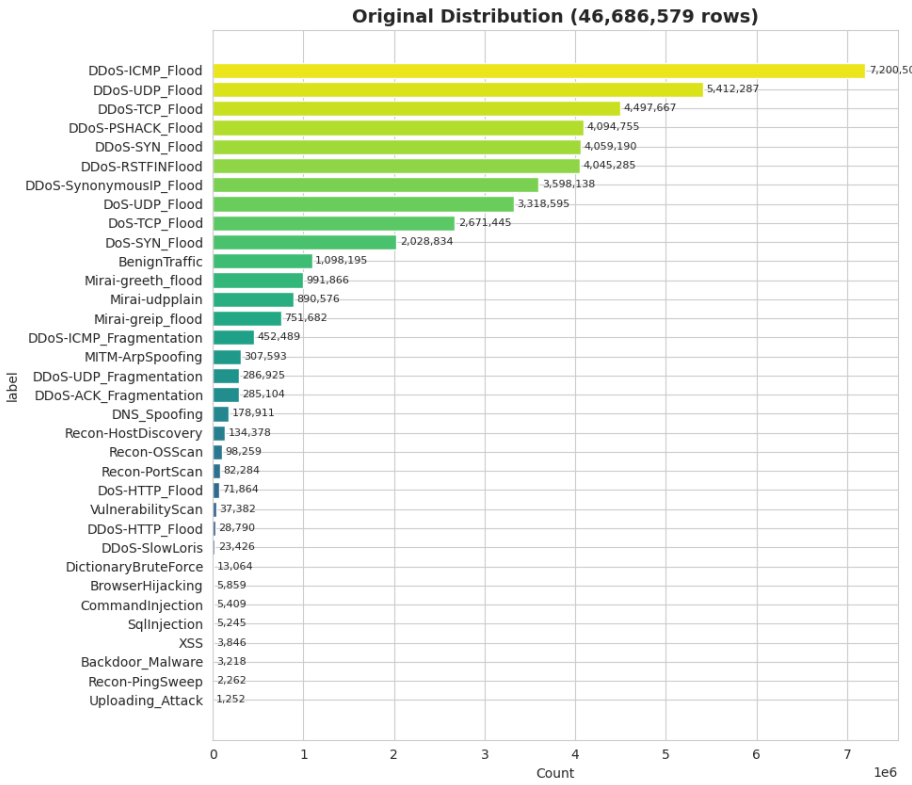

Figure 2 Distribution of Each Class in Original CIC-IOT-23 Dataset

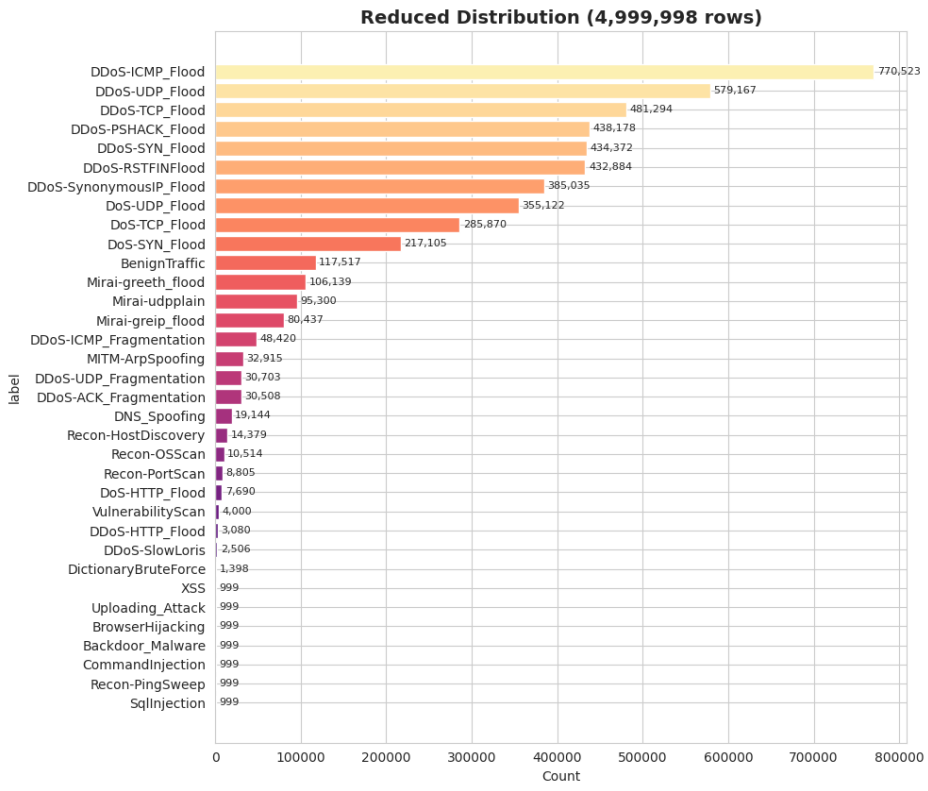

Figure 3 Distribution of Each Class for Undersampled of 34-class Multi-classification Dataset.

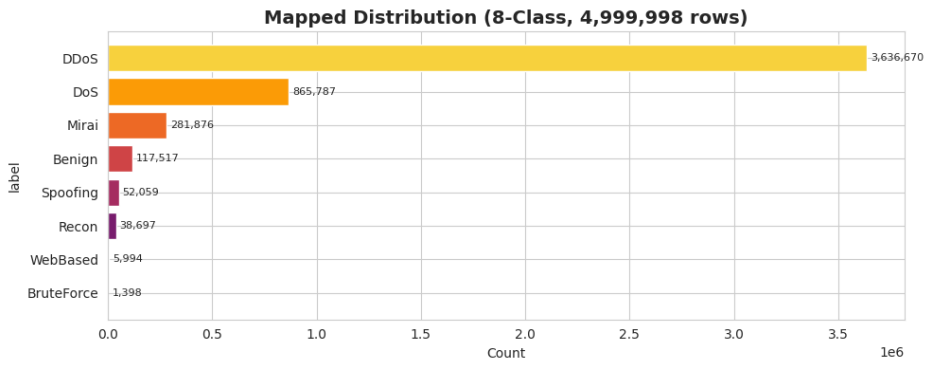

Figure 4 Distribution of Each Class for Undersampled of 8-class Multi-classification Dataset

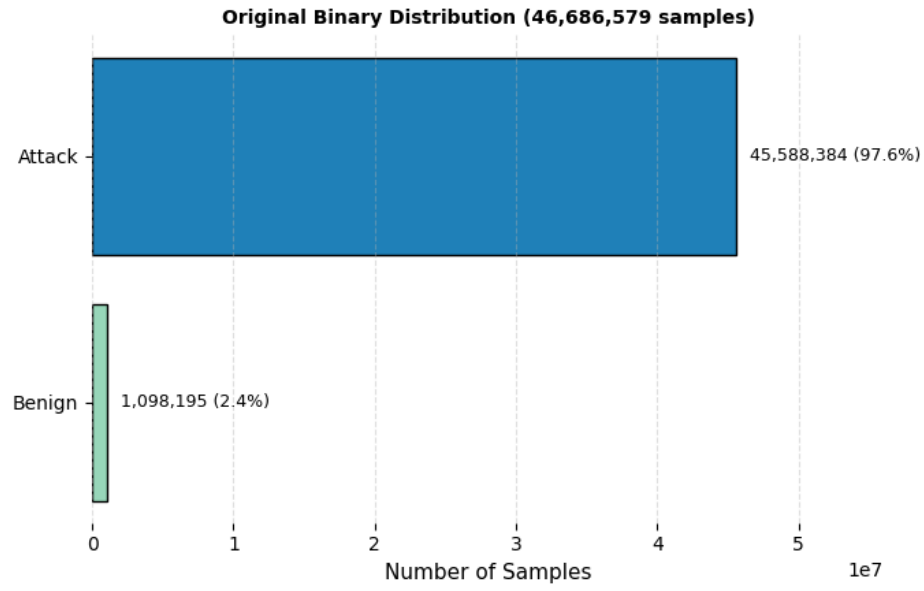

Figure 5 Original Binary Distribution.

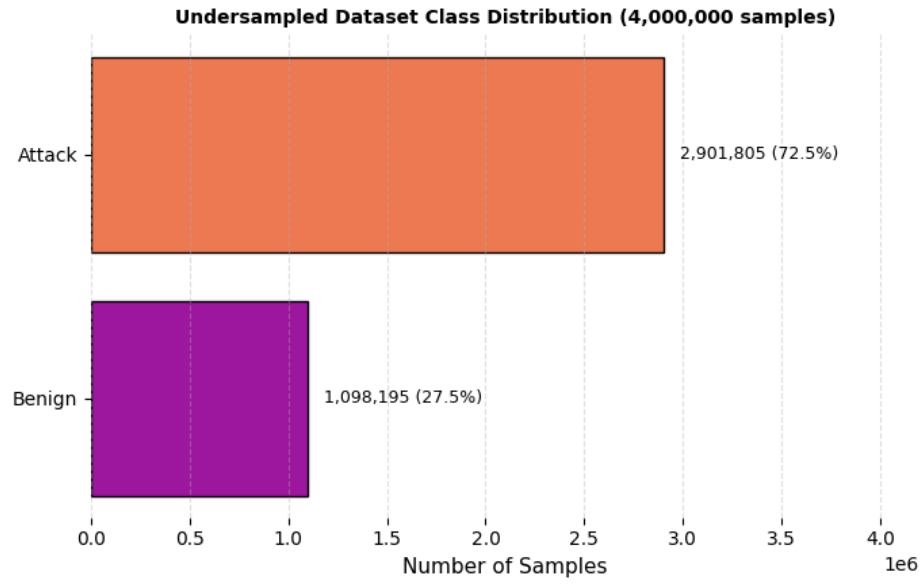

Figure 6 Distribution of each Class for Undersampled Dataset of Binary Classification.

3.2 Data Preprocessing

Given the dataset's scale and imbalance, a structured preprocessing pipeline was implemented as follows[18].

3.2.1 Chunked Data Loading

Due to memory constraints, each CSV file was processed in chunks of 2 million rows. Label frequencies were aggregated across all files to estimate the full distribution before sampling.

3.2.2 Label Mapping and Stratified Sampling

- **Binary Classification:** Original labels were grouped into two categories: Benign and Attack. All benign samples (1,098,195) were retained, and 2,901,805 attack records were randomly sampled, yielding a balanced dataset of 4,000,000 samples. The final binary distribution is shown in (Figure 6).
- **34-Class Multi-Class Classification:** A stratified sampling scheme was used to downsample the full dataset to 5,000,000 rows while maintaining class diversity. A minimum of 1,000 samples per class was enforced to prevent minority class elimination. The resulting distribution is depicted in (Figure 3).
- **8-Class Mapping:** Attack labels were mapped into eight broader categories based on functionality (e.g., all DDoS variants into "DDoS", web-based attacks into "WebBased"). The relabeled and sampled distribution is shown in (Figure 4).

Stratified Undersampling Mathematical Expression:

For each class c_k, we calculate the target number of samples as the following equation:

$$n_k^{\text{target}} = \max\left(M_{\min}, \lfloor \frac{n_k}{N_{\text{orig}}} \times N_{\text{target}} \rfloor\right)$$

Where:

- $C = \{c_1, c_2, \ldots, c_K\}$, set of classes.
- n_k: the original sample count per class c_k
- $N_{\text{orig}} = \sum_k n_k$: The total number of original samples
- N_{target}: The desired total sample size after undersampling
- $M_{\min}$: Minimum allowed samples per class.

3.2.3 Label Encoding and Normalization

For both binary and multi-class scenarios, the mapped labels are encoded as integers. All features are normalized to zero mean and unit variance via standardization.

$$x_j^{\text{std}} = \frac{x_j - \mu_j}{\sigma_j}$$

Where:

- x_j: value of feature j for a given sample
- μ_j: mean of feature j (computed from training set)
- σ_j: standard deviation of feature j (training set)

This normalization enhances numerical stability and accelerates model convergence [19].

3.2.4 Train-Test Splitting

Each dataset variant (binary, 34-class, 8-class) was split into 80% training and 20% testing subsets using stratified sampling to preserve class proportions. These partitions were kept fixed across all model comparisons.

3.3 Feature Selection via LightGBM Gain-Based Importance

To enhance computational efficiency and mitigate overfitting, we employed an embedded feature selection strategy utilizing gain-based importance scores derived from a LightGBM model [20]. Specifically, LightGBM quantifies the importance of each feature by aggregating the total gain defined as the reduction in the objective loss function across all decision tree splits in which the feature participates. Formally, the gain for feature f_j is expressed as:

$$Gain(f_j) = \sum_{splits\ on\ f_j} \Delta Loss$$

Where ΔLoss denotes the decrease in the loss function resulting from splitting on f_j.

Following model training, the gain values for all candidate features are computed, yielding a feature importance vector $\{\text{Gain}(f_1), \ldots, \text{Gain}(f_d)\}$ for d-dimensional input space. We adopt a median-based thresholding approach, retaining only those features with gain values exceeding the median gain across all features:

$$\mathcal{F}_{\text{selected}} = \{f_j \mid \text{Gain}(f_j) > \text{Median}(\{\text{Gain}(f_k)\}_{k=1}^{d})\}$$

This process yields a parsimonious set of high-impact features, substantially reducing input dimensionality while preserving the model's discriminative power. The selected feature subset is subsequently used for all downstream modeling and evaluation tasks. The names and descriptions of the selected features are summarized in Table 6. The ranking and selection outcome are illustrated in (Figure 27), which depicts the top features as determined by their gain-based importance.

3.4 Classification Pipelines

Two classification pipelines were developed: one for binary classification (Benign vs. Attack) and another for multi-class classification (34 classes and 8 attack families). Both pipelines follow the workflow illustrated in Figure 1, with shared preprocessing and feature selection components.

3.4.1 Binary Classification Pipeline

The binary pipeline evaluates the ability to distinguish malicious traffic from benign traffic. The following models were included:

- **Gradient Boosting Models**: XGBoost, LightGBM, CatBoost, NGBoost
- **Ensemble and Linear Models**: ExtraTrees, Logistic Regression, AdaBoost
- **Classical Baselines**: Decision Tree, Random Forest, Linear SVM, K-Nearest Neighbors, Gaussian Naive Bayes

To address residual class imbalance, class weights were assigned using:

$$w_i = \frac{N}{k \cdot n_i}$$

where N is the dataset size, $k = 2$ (binary classes), and n_i is the sample count for class i.

Hyperparameters for all models were optimized via grid search using five-fold cross-validation on the training data [21], targeting macro F1-score. Final models were retrained on the full training set and evaluated on the holdout test split.

3.4.2 Multi-class Classification Pipeline

or the 34-class and 8-class classification tasks, we benchmarked three tree-based gradient boosting frameworks:

- **XGBoost**
- **LightGBM**
- **CatBoost**

Key configurations included:

- **Sample Weighting**: Weights were inversely proportional to class frequencies to mitigate imbalance. Which can be calculated from the following equation:
  - The class weight for class k:

    $$w_k = \frac{N}{K \cdot n_k}$$

    Where:

    - N: the number of training samples.
    - K: the number of classes.
    - n_k: the number of training samples in class k.
- **Early Stopping**: Training was halted if no improvement in validation loss was observed over 20–30 iterations.
- **Label Encoding**: All class labels were mapped to integer values; CatBoost handled categorical features natively.

All models were trained using the same 23-feature subset selected via LightGBM gain, ensuring fair comparison across tasks.

3.4.2 Evaluation Metrics

Model performance is assessed using the following standard metrics:

- **Accuracy:** Overall correctness of predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP, TN, FP and FN denote true positives, true negatives, false positives, and false negatives, respectively.

- **Precision, Recall, F1-score:** Computed per class; macro and weighted averages were used for multi-class scenarios.

For class i:

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \text{Recall}_i = \frac{TP_i}{TP_i + FN_i}$$

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

- **Confusion matrix:** Provided to analyze misclassifications and class confusion patterns.
- **Inference latency:** Computed as both total prediction time and average time per sample, using the entire test set. Latency was measured in milliseconds or microseconds using high-resolution timing utilities, reflecting realistic operational performance [22].

4. Experimental Results

4.1 Experimental Setup

All experiments were conducted on a server configured with the specifications listed in Table 2. Training was executed using Python 3.11 with support for GPU-accelerated libraries including XGBoost (v2.0.3), LightGBM, and CatBoost. GPU support was provided by an NVIDIA Tesla P100 (16GB). Hyperparameters for all models were selected via grid search with stratified 5-fold cross-validation, optimizing for macro F1-score on the training set.

Table 2 Equipment Specifications and Environment Settings

Project	**Properties**
OS	Linux-6.6.56+-x86_64-with-glibc2.35
CPU	Intel(R) Xeon(R) CPU @ 2.00GHz
GPU	Tesla P100-PCIE-16GB (16384 MiB)
Memory	31.35 GB RAM
Disk	57 GB
Python	3.11.11
NVIDIA CUDA	11.8
Framework	TensorFlow 2.18.0 / Torch 2.6.0+cu124 / XGBoost 2.0.3

4.2 Binary Classification Results

Table 3 summarizes the performance of advanced learners and baseline algorithms on the binary intrusion detection task.

Table 3 Result of Gradient Boosting Algorithm of Binary classification

Model	Accuracy	Macro F1	Precision	Recall	Inference Latency (ms/sampl
XGBoost	**99.62**	**0.99**	**0.99**	**1.00**	**0.0004**
NGBoost	99.58	0.99	0.99	1.00	0.0276
CatBoost	99.57	0.99	0.99	1.00	0.0080
LightGBM	99.55	0.99	0.99	1.00	0.0052
ExtraTrees	98.92	0.99	0.98	0.99	0.0050
AdaBoost	98.12	0.98	0.97	0.98	0.0039
Random Forest	99.55	0.99	0.99	1.00	0.0014
Decision Tree	99.49	0.99	0.99	1.00	0.0001
KNN	99.02	0.99	0.98	0.99	0.2502
Logistic Regression	98.22	0.98	0.97	0.98	0.0001
Linear SVM	98.21	0.98	0.97	0.99	0.0001

The binary classification experiments distinguish benign from attack traffic on the CIC-IoT-2023 dataset using seven advanced learners and three baseline algorithms. As shown in Table 3, all gradient boosting models substantially outperform traditional methods, with XGBoost achieving the highest accuracy of 99.62 % and a macro-F_1 score of 0.99. Notably, XGBoost also exhibits the lowest inference latency (0.0004 ms/sample), underscoring its suitability for real-time deployment. NGBoost, CatBoost, and LightGBM follow closely, each exceeding 99.5 % accuracy and maintaining latencies below 0.03 ms/sample. In contrast, classical classifiers such as Logistic Regression and Linear SVM, while still achieving respectable accuracies near 98.2 %, incur marginally higher error rates and are thus less favorable for high-assurance intrusion detection.

A detailed examination of the classification reports (Figures 6–16) reveals consistently high precision and recall for both classes across the boosting models. For instance, XGBoost (Figure 7) attains precision and recall of 0.99 and 1.00, respectively, for benign traffic and perfect scores for attack traffic, yielding an overall F_1 of 0.995. LightGBM and CatBoost display virtually identical behavior, with macro-F_1 and weighted-F_1 values of 0.99 and 1.00 (Figures 8–9). Even baseline ensembles, such as Random Forest (Figure 14) and ExtraTrees (Figure 11), maintain macro-F_1 scores above 0.98 with minimal degradation. Conversely, simpler models like k-NN (Figure 16) exhibit slight increases in false negatives, reflected by a lower recall for benign samples (0.99) and a corresponding dip in F_1 to 0.99.

The confusion matrices further corroborate these findings by illustrating that boosting algorithms produce near-perfect separation between benign and attack classes. In XGBoost's confusion matrix (Figure 7), off-diagonal entries are virtually zero, indicating negligible false positives or negatives. Similar patterns emerge for LightGBM and CatBoost (Figures 8–9), where misclassifications are confined to a handful of borderline instances. Traditional baselines, such as Decision Tree (Figure 12) and Logistic Regression (Figure 13), show marginally higher misclassification rates, primarily reflecting occasional false negatives that could be critical in operational settings. k-NN and Linear SVM (Figures 15–16) present the most dispersed error patterns, highlighting the intrinsic limitations of distance-based and linear decision boundaries in this context.

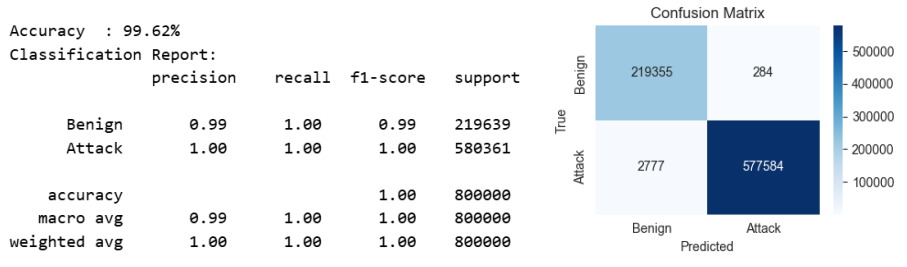

Figure 7 Classification Report & Confusion Matrix of XGBoost For Binary Classification

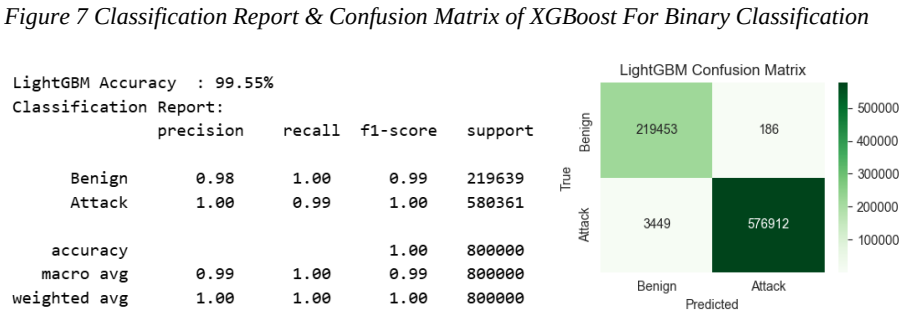

Figure 8 Classification Report & Confusion Matrix of LGB For Binary Classification

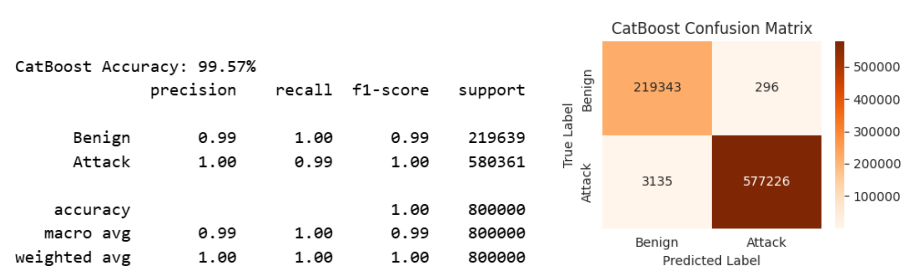

Figure 9 Classification Report & Confusion Matrix of CatBoost For Binary Classification

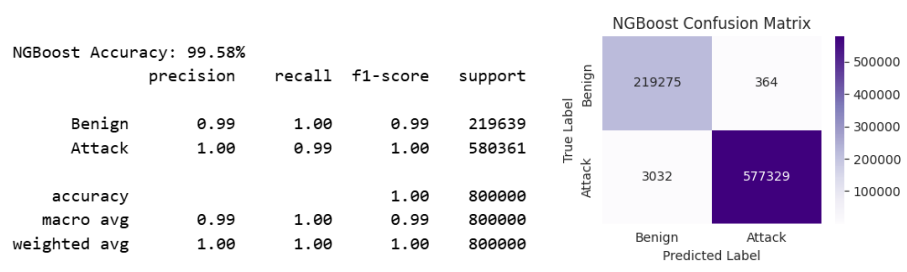

Figure 10 Classification Report & Confusion Matrix of NGBoost For Binary Classification

ExtraTrees Accuracy: 98.92%

	precision	recall	f1-score	support
Benign	0.96	1.00	0.98	219639
Attack	1.00	0.99	0.99	580361
accuracy			0.99	800000
macro avg	0.98	0.99	0.99	800000
weighted avg	0.99	0.99	0.99	800000

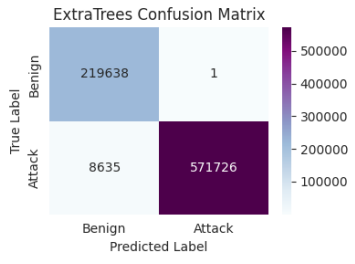

Figure 11 Classification Report & Confusion Matrix of ExtraTrees for Binary Classification

Decision Tree Accuracy: 99.49%

	precision	recall	f1-score	support
Benign	0.98	1.00	0.99	219639
Attack	1.00	0.99	1.00	580361
accuracy			0.99	800000
macro avg	0.99	1.00	0.99	800000
weighted avg	0.99	0.99	0.99	800000

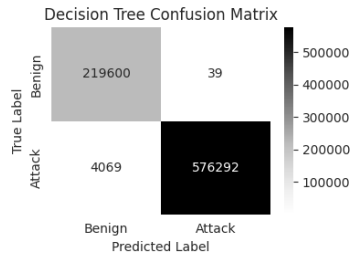

Figure 12 Classification Report & Confusion Matrix of Decision Tree for Binary Classification

LogisticRegression Accuracy: 98.22%

	precision	recall	f1-score	support
Benign	0.95	0.99	0.97	219639
Attack	1.00	0.98	0.99	580361
accuracy			0.98	800000
macro avg	0.97	0.98	0.98	800000
weighted avg	0.98	0.98	0.98	800000

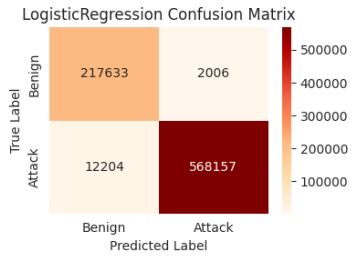

Figure 13 Classification Report & Confusion Matrix of Logistic Regression for Binary Classification

Random Forest Accuracy: 99.55%

	precision	recall	f1-score	support
Benign	0.98	1.00	0.99	219639
Attack	1.00	0.99	1.00	580361
accuracy			1.00	800000
macro avg	0.99	1.00	0.99	800000
weighted avg	1.00	1.00	1.00	800000

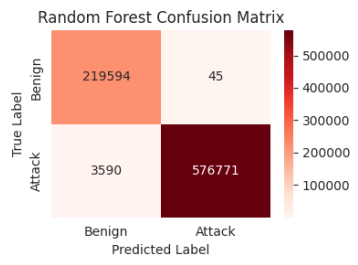

Figure 14 Classification Report & Confusion Matrix of Random Forest for Binary Classification

Linear SVM (SGD) Accuracy: 98.21%

	precision	recall	f1-score	support
Benign	0.95	0.99	0.97	219639
Attack	1.00	0.98	0.99	580361
accuracy			0.98	800000
macro avg	0.97	0.99	0.98	800000
weighted avg	0.98	0.98	0.98	800000

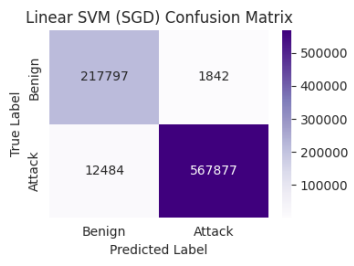

Figure 15 Classification Report & Confusion Matrix of Linear SVM for Binary Classification

k-NN (subsampled) Accuracy: 99.02%

	precision	recall	f1-score	support
Benign	0.97	1.00	0.98	219639
Attack	1.00	0.99	0.99	580361
accuracy			0.99	800000
macro avg	0.98	0.99	0.99	800000
weighted avg	0.99	0.99	0.99	800000

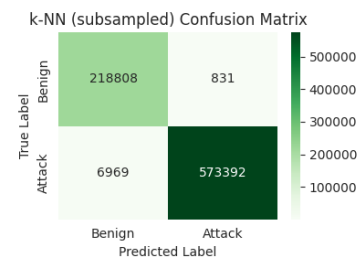

Figure 16 Classification Report & Confusion Matrix of KNN for Binary Classification

AdaBoost Accuracy: 98.12%

	precision	recall	f1-score	support
Benign	0.95	0.99	0.97	219639
Attack	0.99	0.98	0.99	580361
accuracy			0.98	800000
macro avg	0.97	0.98	0.98	800000
weighted avg	0.98	0.98	0.98	800000

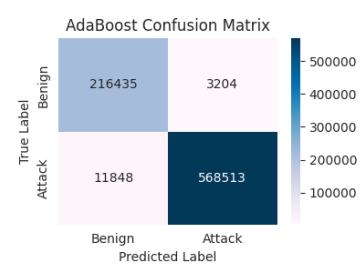

Figure 17 Classification Report & Confusion Matrix of AdaBoost for Binary Classification

In summary, the binary classification results demonstrate that LightGBM gain-based feature selection and gradient boosting ensembles yield state-of-the-art performance on the CIC-IoT-2023 dataset. Table 3 quantifies the superiority of these models in accuracy, macro-F_1, and inference latency, while Figures 7–17 validate their robustness through classification reports and confusion matrices. Collectively, these results establish XGBoost and its boosting counterparts as highly effective and efficient solutions for real-time IoT intrusion detection.

4.3 Multi-Class Classification Results (34 Classes)

Table 4 presents the results for the multi-class intrusion detection scenario, where traffic is still as it is 34 class:

Table 4 Result of Gradient Boosting Algorithms for Multi classification

Model	Accuracy	**Weighted F1**	Macro F1	Precision	Recall	Inference (ms/sample)
XGBoost	**99.454**	**0.99**	**0.88**	**0.91**	**0.87**	**0.0069**
LightGBM	98.946	0.99	0.76	0.74	0.80	0.3120
CatBoost	98.839	0.99	0.74	0.73	0.75	0.0157

The multi-class evaluation extends the binary framework to discriminate among benign traffic and 33 distinct attack types, organized into eight aggregated families. As summarized in Table 4, XGBoost again leads the ensemble methods with an overall accuracy of 99.45 % and a weighted F_1 score of 0.99. LightGBM and CatBoost follow with accuracies of 98.95 % and 98.84 %, respectively, while maintaining weighted F_1 scores above 0.99. Despite the increased task complexity, inference latencies remain sub-millisecond, with XGBoost predicting at an average of 0.0069 ms per sample. These results confirm that gradient boosting models, when combined with LightGBM gain-based feature selection, effectively scale from binary to rich multi-class scenarios without sacrificing real-time performance.

A closer inspection of the class-wise metrics (Figures 18–20) highlights uniformly high precision and recall across most categories. For example, XGBoost achieves per-class F_1 scores exceeding 0.99 for major families such as DDoS-ICMP_Flood and DoS-TCP_Flood, with only slight degradation ($F_1 \approx 0.93$) observed in minority classes like BrowserHijacking and XSS. LightGBM (Figure 20) and CatBoost (Figure 22) display similar robustness, with macro-F_1 scores of 0.88 and 0.74, respectively, reflecting the inherent difficulty of balancing many classes. In all cases, weighted averages (precision, recall, and F_1) remain at or above 0.99, demonstrating that the most frequent classes dominate overall performance without entirely obscuring minority-class behavior.

XGBoost Accuracy: 0.994543

	precision	recall	f1-score	support
Backdoor_Malware	0.73	0.59	0.65	200
BenignTraffic	0.94	0.95	0.94	23503
BrowserHijacking	0.87	0.62	0.73	200
CommandInjection	0.84	0.55	0.66	200
DDoS-ACK_Fragmentation	1.00	1.00	1.00	6101
DDoS-HTTP_Flood	1.00	0.99	0.99	616
DDoS-ICMP_Flood	1.00	1.00	1.00	154105
DDoS-ICMP_Fragmentation	1.00	1.00	1.00	9684
DDoS-PSHACK_Flood	1.00	1.00	1.00	87636
DDoS-RSTFINFlood	1.00	1.00	1.00	86577
DDoS-SYN_Flood	1.00	1.00	1.00	86874
DDoS-SlowLoris	0.99	0.99	0.99	501
DDoS-SynonymousIP_Flood	1.00	1.00	1.00	77007
DDoS-TCP_Flood	1.00	1.00	1.00	96259
DDoS-UDP_Flood	1.00	1.00	1.00	115833
DDoS-UDP_Fragmentation	1.00	1.00	1.00	6141
DNS_Spoofing	0.74	0.79	0.77	3829
DictionaryBruteForce	0.77	0.59	0.67	279
DoS-HTTP_Flood	1.00	1.00	1.00	1538
DoS-SYN_Flood	1.00	1.00	1.00	43421
DoS-TCP_Flood	1.00	1.00	1.00	57174
DoS-UDP_Flood	1.00	1.00	1.00	71024
MITM-ArpSpoofing	0.90	0.86	0.88	6583
Mirai-greeth_flood	1.00	1.00	1.00	21228
Mirai-greip_flood	1.00	1.00	1.00	16087
Mirai-udpplain	1.00	1.00	1.00	19060
Recon-HostDiscovery	0.82	0.87	0.85	2876
Recon-OSScan	0.70	0.69	0.69	2103
Recon-PingSweep	0.75	0.58	0.66	200
Recon-PortScan	0.67	0.74	0.70	1761
SqlInjection	0.89	0.54	0.67	200
Uploading_Attack	0.77	0.50	0.61	200
VulnerabilityScan	0.99	1.00	1.00	800
XSS	0.67	0.58	0.62	200
accuracy			0.99	1000000
macro avg	0.91	0.87	0.88	1000000
weighted avg	0.99	0.99	0.99	1000000

Figure 18 Classification Report of XGBoost For Multi Classification

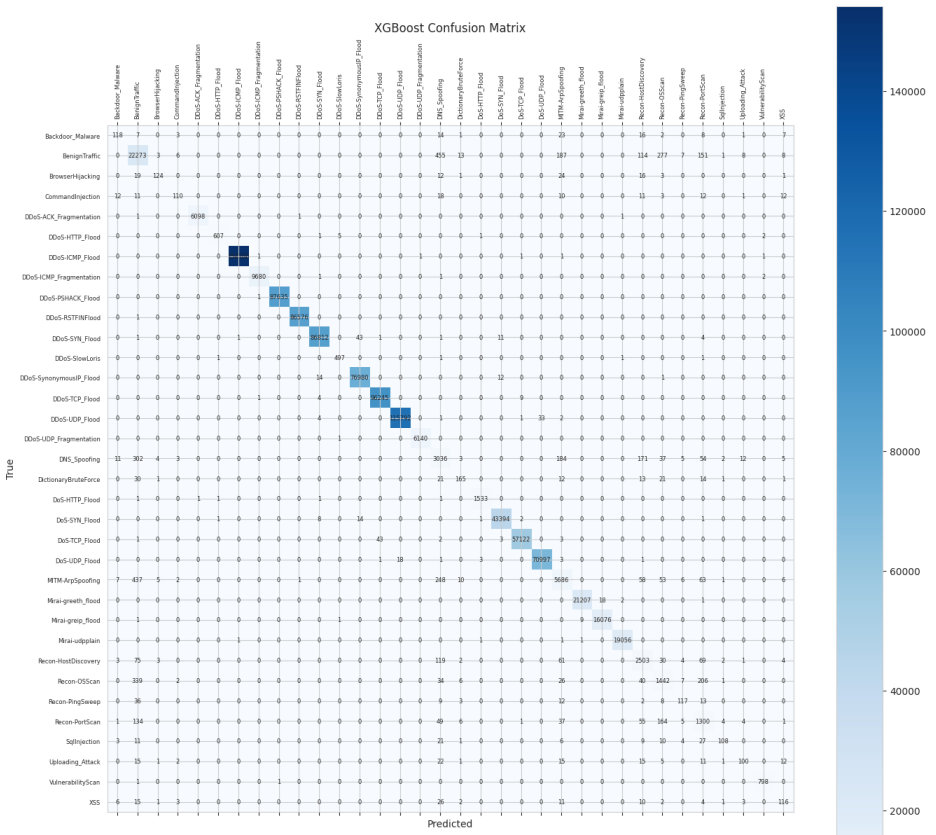

Figure 19 Confusion Matrix of XGBoost For Multi Classification

LightGBM Accuracy: 0.989463

	precision	recall	f1-score	support
Backdoor_Malware	0.12	0.23	0.16	200
BenignTraffic	0.96	0.82	0.88	23503
BrowserHijacking	0.22	0.46	0.30	200
CommandInjection	0.20	0.29	0.24	200
DDoS-ACK_Fragmentation	1.00	1.00	1.00	6101
DDoS-HTTP_Flood	1.00	0.98	0.99	616
DDoS-ICMP_Flood	1.00	1.00	1.00	154105
DDoS-ICMP_Fragmentation	1.00	1.00	1.00	9684
DDoS-PSHACK_Flood	1.00	1.00	1.00	87636
DDoS-RSTFINFlood	1.00	1.00	1.00	86577
DDoS-SYN_Flood	1.00	1.00	1.00	86874
DDoS-SlowLoris	0.94	0.99	0.96	501
DDoS-SynonymousIP_Flood	1.00	1.00	1.00	77007
DDoS-TCP_Flood	1.00	1.00	1.00	96259
DDoS-UDP_Flood	1.00	1.00	1.00	115833
DDoS-UDP_Fragmentation	1.00	1.00	1.00	6141
DNS_Spoofing	0.63	0.67	0.65	3829
DictionaryBruteForce	0.21	0.54	0.30	279
DoS-HTTP_Flood	0.99	1.00	0.99	1538
DoS-SYN_Flood	1.00	1.00	1.00	43421
DoS-TCP_Flood	1.00	1.00	1.00	57174
DoS-UDP_Flood	1.00	1.00	1.00	71024
MITM-ArpSpoofing	0.86	0.79	0.82	6583
Mirai-greeth_flood	1.00	1.00	1.00	21228
Mirai-greip_flood	1.00	1.00	1.00	16087
Mirai-udpplain	1.00	1.00	1.00	19060
Recon-HostDiscovery	0.75	0.78	0.76	2876
Recon-OSScan	0.36	0.52	0.42	2103
Recon-PingSweep	0.12	0.42	0.19	200
Recon-PortScan	0.58	0.69	0.63	1761
SqlInjection	0.14	0.45	0.21	200
Uploading_Attack	0.15	0.29	0.20	200
VulnerabilityScan	0.97	1.00	0.98	800
XSS	0.09	0.23	0.13	200
accuracy			0.99	1000000
macro avg	0.74	0.80	0.76	1000000
weighted avg	0.99	0.99	0.99	1000000

Figure 20 Classification Report of LGB For Multi Classification

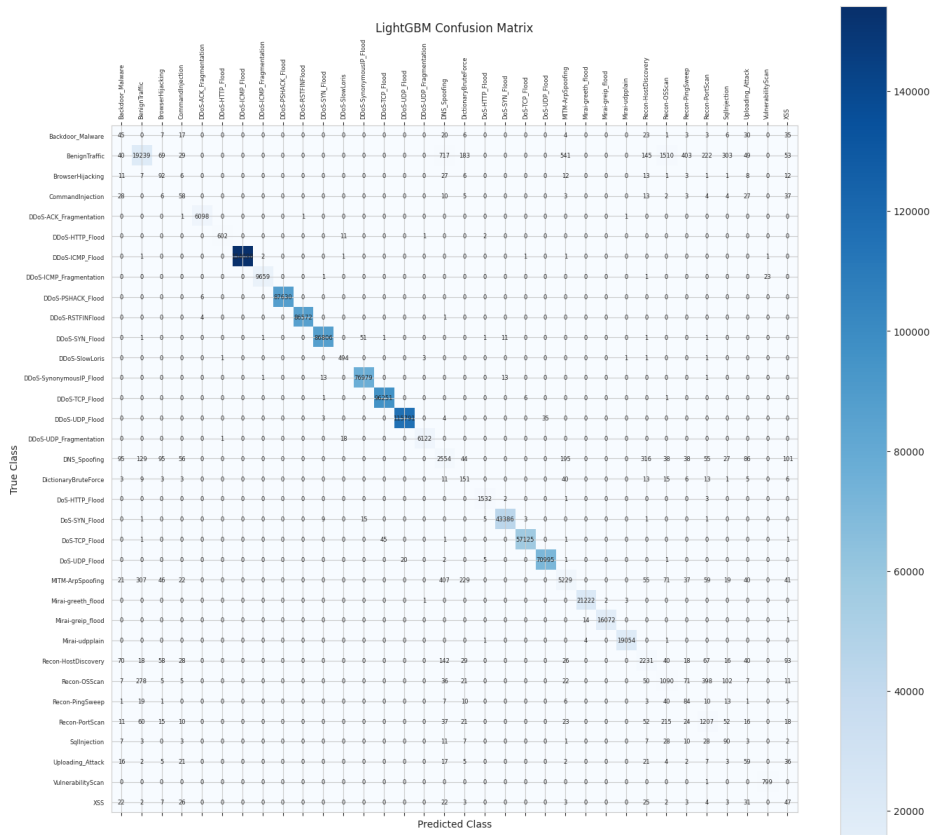

Figure 21 Confusion Matrix of LGB For Multi Classification

The confusion matrices (Figures 19, 21, 23) further elucidate model behavior by revealing that misclassifications are scattered rather than concentrated in a single category. XGBoost's confusion matrix (Figure 19) shows minimal cross-family errors, with isolated confusions primarily between Recon-PortScan and Recon-PingSweep two classes sharing similar flow characteristics. LightGBM and CatBoost exhibit comparable patterns, underscoring the effectiveness of the selected 23 features in preserving inter-class distinctions. Crucially, no systematic bias toward benign or attack families emerges, thereby ensuring that rare attack types are not consistently overlooked.

CatBoost Accuracy: 0.988393

	precision	recall	f1-score	support
Backdoor_Malware	0.15	0.10	0.12	200
BenignTraffic	0.94	0.85	0.89	23503
BrowserHijacking	0.31	0.27	0.29	200
CommandInjection	0.27	0.20	0.23	200
DDoS-ACK_Fragmentation	0.99	0.99	0.99	6101
DDoS-HTTP_Flood	0.95	0.98	0.96	616
DDoS-ICMP_Flood	1.00	1.00	1.00	154105
DDoS-ICMP_Fragmentation	0.99	0.99	0.99	9684
DDoS-PSHACK_Flood	1.00	1.00	1.00	87636
DDoS-RSTFINFlood	1.00	1.00	1.00	86577
DDoS-SYN_Flood	1.00	1.00	1.00	86874
DDoS-SlowLoris	0.83	0.98	0.90	501
DDoS-SynonymousIP_Flood	1.00	1.00	1.00	77007
DDoS-TCP_Flood	1.00	1.00	1.00	96259
DDoS-UDP_Flood	1.00	1.00	1.00	115833
DDoS-UDP_Fragmentation	0.99	0.99	0.99	6141
DNS_Spoofing	0.54	0.65	0.59	3829
DictionaryBruteForce	0.35	0.35	0.35	279
DoS-HTTP_Flood	0.94	0.99	0.97	1538
DoS-SYN_Flood	1.00	1.00	1.00	43421
DoS-TCP_Flood	1.00	1.00	1.00	57174
DoS-UDP_Flood	1.00	1.00	1.00	71024
MITM-ArpSpoofing	0.79	0.75	0.77	6583
Mirai-greeth_flood	1.00	1.00	1.00	21228
Mirai-greip_flood	1.00	1.00	1.00	16087
Mirai-udpplain	1.00	1.00	1.00	19060
Recon-HostDiscovery	0.69	0.78	0.73	2876
Recon-OSScan	0.33	0.46	0.39	2103
Recon-PingSweep	0.21	0.23	0.22	200
Recon-PortScan	0.40	0.54	0.46	1761
SqlInjection	0.22	0.21	0.22	200
Uploading_Attack	0.19	0.25	0.21	200
VulnerabilityScan	0.83	0.99	0.90	800
XSS	0.10	0.10	0.10	200
accuracy			0.99	1000000
macro avg	0.73	0.75	0.74	1000000
weighted avg	0.99	0.99	0.99	1000000

Figure 22 Classification Report of CatBoost For Multi Classification

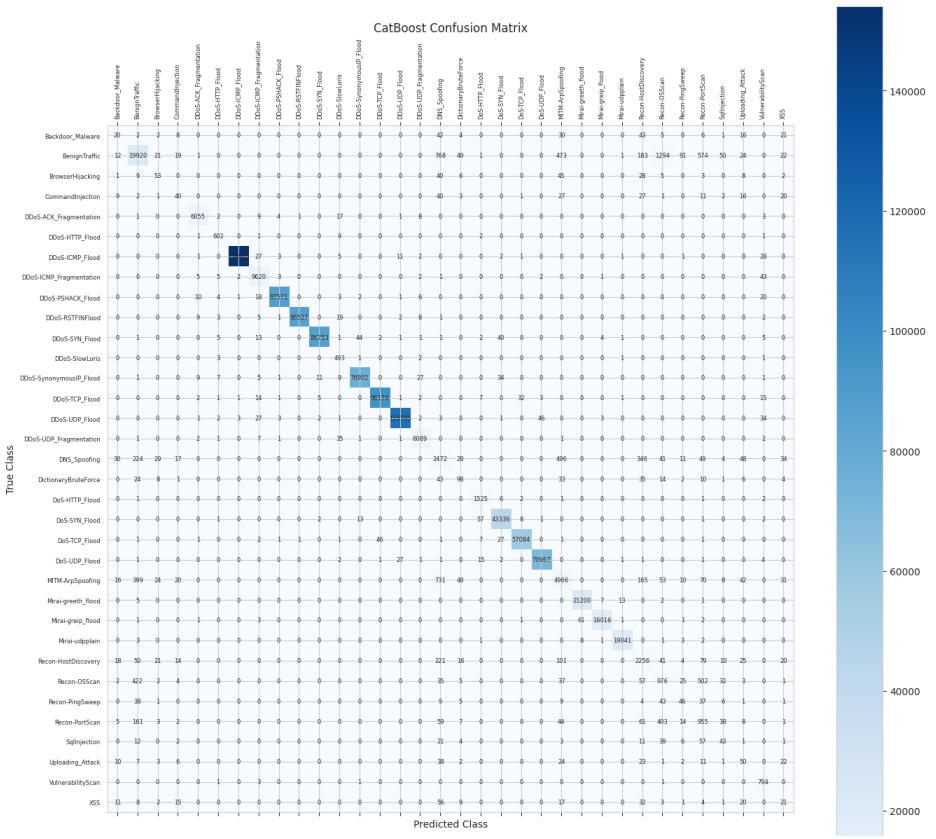

Figure 23 Confusion Matrix of CatBoost For Multi Classification

Taken together, the multi-class results affirm that gradient boosting ensembles strike an optimal balance between classification breadth and operational speed. Table 4 quantifies the high accuracy and sub-millisecond inference times achieved across all models, while Figures 17–22 validate their capacity to handle class imbalance and maintain high per-class fidelity. These findings reinforce the suitability of the proposed dual-pipeline framework for comprehensive, real-time intrusion detection in heterogeneous IoT environments.

4.4 Multi-Class Classification Results (8 Classes)

Table 5 presents the results for the multi-class intrusion detection scenario, where traffic is categorized into eight classes (benign + seven attack families:

Table 5 Result of Gradient Boosting Algorithms for 8-class Multi classification

Model	Accuracy	**Weighted F1**	Macro F1	Precision	Recall	Inference (ms/sample)
XGBoost	**99.58**	**1.00**	**0.89**	**0.90**	**0.88**	**0.0015**
LightGBM	99.38	0.99	0.80	0.78	0.82	0.1935
CatBoost	99.25	0.99	0.77	0.78	0.78	0.0099

Table 5 reports the performance of XGBoost, LightGBM, and CatBoost on the eight-class intrusion detection task (benign + seven attack families). XGBoost again achieves the highest overall accuracy of 99.58 % and perfect weighted F_1 of 1.00, with a macro F_1 of 0.89. Its per-class precision (0.90) and recall (0.88) remain uniformly high, despite the increased number of target labels, while maintaining an ultra-low inference latency of 0.0015 ms/sample. LightGBM attains 99.38 % accuracy and weighted F_1 of 0.99, but exhibits a lower macro F_1 of 0.80, reflecting modest degradation on the less frequent classes; its precision and recall (0.78 and 0.82, respectively) indicate that some minority families are more challenging to distinguish. CatBoost follows with 99.25 % accuracy, weighted F_1 of 0.99, and macro F_1 of 0.77, yielding balanced precision and recall of 0.78 across classes and an inference time of 0.0099 ms/sample.

The classification reports and confusion matrices in Figures 24–26 further illuminate per-class performance. In Figure 23, XGBoost's report shows that major families such as DDoS-ICMP_Flood and DoS-TCP_Flood achieve near-perfect F_1 scores, while smaller classes like Recon and Malware variants maintain F_1 above 0.85. LightGBM's report (Figure 25) confirms strong detection of dominant classes but reveals lower recall for families with limited samples consistent with its macro F_1 of 0.80. CatBoost (Figure 26) similarly sustains high precision across most classes but records slightly more false negatives in underrepresented categories.

The corresponding confusion matrices highlight that misclassifications are neither pervasive nor systematically biased toward any single family.

XGBoost (Figure 24) mislabels only a handful of samples, primarily confusing closely related reconnaissance attacks. LightGBM (Figure 25) and CatBoost (Figure 26) exhibit comparable sparse off-diagonal patterns, indicating that errors occur mainly between attack families sharing similar network-flow characteristics. Importantly, no significant confusion arises between benign and attack groups, confirming the robustness of all three ensembles.

XGBoost Accuracy: 0.995828

	precision	recall	f1-score	support
Benign	0.94	0.95	0.95	23503
BruteForce	0.81	0.62	0.70	280
DDoS	1.00	1.00	1.00	727334
DoS	1.00	1.00	1.00	173158
Mirai	1.00	1.00	1.00	56375
Recon	0.85	0.86	0.86	7739
Spoofing	0.91	0.87	0.89	10412
WebBased	0.66	0.71	0.68	1199
accuracy			1.00	1000000
macro avg	0.90	0.88	0.89	1000000
weighted avg	1.00	1.00	1.00	1000000

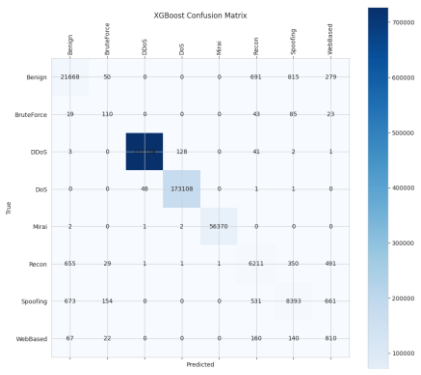

Figure 24 Classification Report & Confusion Matrix of XGBoost for 8-class Classification

LightGBM Accuracy: 0.993829

	precision	recall	f1-score	support
Benign	0.94	0.92	0.93	23503
BruteForce	0.30	0.39	0.34	280
DDoS	1.00	1.00	1.00	727334
DoS	1.00	1.00	1.00	173158
Mirai	1.00	1.00	1.00	56375
Recon	0.81	0.80	0.81	7739
Spoofing	0.86	0.81	0.83	10412
WebBased	0.36	0.68	0.47	1199
accuracy			0.99	1000000
macro avg	0.78	0.82	0.80	1000000
weighted avg	0.99	0.99	0.99	1000000

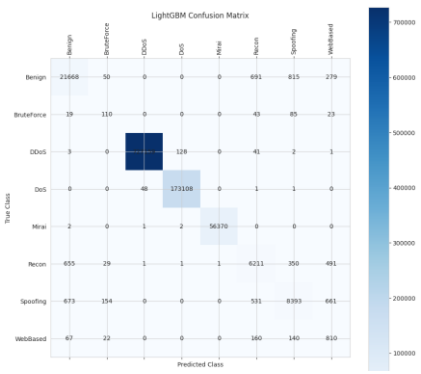

Figure 25 Classification Report & Confusion Matrix of LGB for 8-class Classification

CatBoost Accuracy: 0.992575

	precision	recall	f1-score	support
Benign	0.92	0.90	0.91	23503
BruteForce	0.37	0.24	0.29	280
DDoS	1.00	1.00	1.00	727334
DoS	1.00	1.00	1.00	173158
Mirai	1.00	1.00	1.00	56375
Recon	0.71	0.77	0.74	7739
Spoofing	0.83	0.82	0.82	10412
WebBased	0.38	0.49	0.42	1199
accuracy			0.99	1000000
macro avg	0.78	0.78	0.77	1000000
weighted avg	0.99	0.99	0.99	1000000

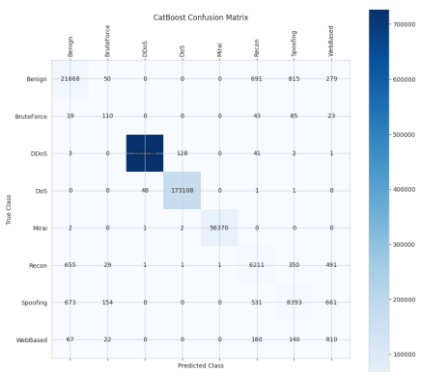

Figure 26 Classification Report & Confusion Matrix of CatBoost for 8-class Classification

Overall, the eight-class results demonstrate that gradient boosting models retain excellent discrimination power even as the classification granularity increases. Table 5 quantifies their superior accuracy and sub-microsecond inference, while Figures 24–26 validate high per-class fidelity and minimal misclassification. These findings confirm that the proposed multi-class pipeline generalizes effectively from binary and 34-class settings to an operational eight-class taxonomy, preserving both performance and real-time applicability.

4.5 Feature Importance and Model Interpretation

To better understand the decision-making process of the gradient boosting models, we analyzed feature importance using the gain metric provided by LightGBM. This metric quantifies the cumulative reduction in the loss function achieved by each feature across all splits in the ensemble, thereby identifying the most discriminative inputs contributing to model performance.

The top 23 features selected through LGB's gain-based selection process are visualized in Figure 27, which presents their relative importance scaled to a 0–100 range. The features were extracted from an initial pool of 46 and used consistently across all experiments for binary, 8-class, and 34-class classification tasks.

Among the most influential features, inter-arrival time (IAT) emerged as the dominant predictor, followed by Rate, Header_Length, and flow_duration. These features collectively capture the temporal and structural properties of network flows—characteristics often altered during malicious activities such as flooding or scanning attacks.

Other features with substantial contributions include:

- Variance and Covariance, which reflect statistical irregularities within flows.

- TCP flag counts such as urg_count, syn_count, and rst_count, indicating specific packet behaviors linked to attack signatures.
- Protocol Type and UDP/TCP indicators, which offer insight into the transport-level context of the flow.
- Aggregated statistics such as Max, Min, Tot size, and Magnitude, which help distinguish between short bursts and long persistent flows.

In contrast, features such as fin_flag_number, fin_count, and ack_count contributed relatively little to the models' decisions, as indicated by their low gain values.

This ranking confirms that temporal dynamics (e.g., IAT, flow_duration) and flow-based statistical aggregates play a central role in detecting anomalies in IoT traffic. Their prominence is consistent with prior studies that emphasize the importance of timing patterns and traffic volume variations in distinguishing between benign and attack behavior.

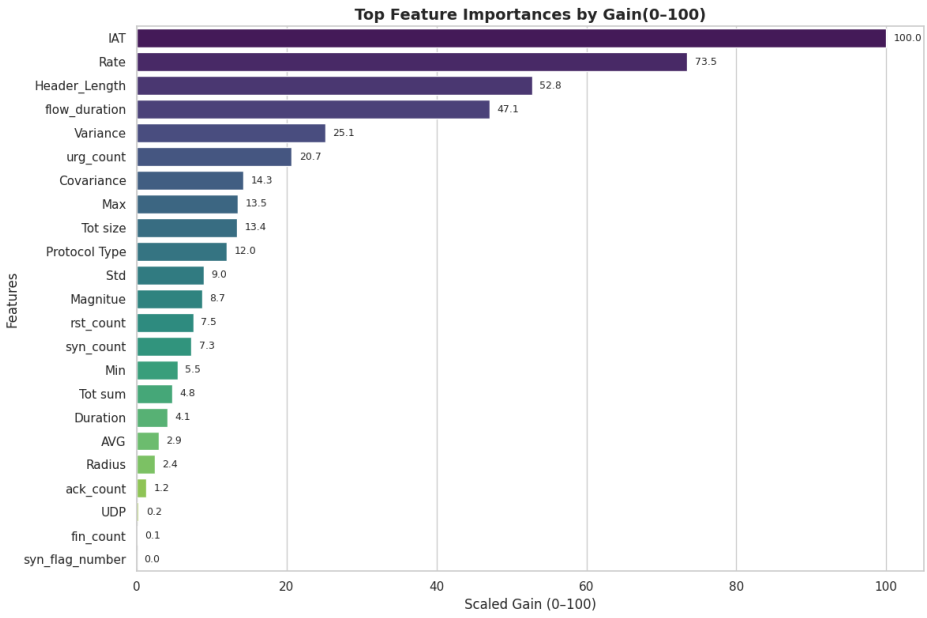

Figure 27 Top 23 Features Ranked by LightGBM Gain-Based Importance

Table 6 Extracted Features 23/46

#	Feature Name	Description
1	flow_duration	Total time duration of the network flow
2	Header_Length	Combined length of packet headers
3	Protocol Type	Communication protocol used (e.g., TCP, UDP, ICMP)
4	Duration	Duration of connection or application-level exchange
5	Rate	Overall transmission rate (e.g., packets/sec or bytes/sec)
6	ack_count	Count of acknowledgement (ACK) packets
7	syn_count	Count of synchronization (SYN) packets
8	fin_count	Count of finish (FIN) packets
9	urg_count	Count of urgent (URG) packets
10	rst_count	Count of reset (RST) packets
11	HTTP	Binary indicator for HTTP traffic
12	TCP	Binary indicator for TCP transport layer
13	Tot sum	Sum of numerical values across some traffic attribute
14	Min	Minimum value in a given statistical measurement window
15	Max	Maximum value observed in a traffic-related feature
16	AVG	Arithmetic mean across packets or flows
17	Std	Standard deviation indicating spread or variability
18	Tot size	Total size of packets in a flow
19	IAT	Inter-arrival time between packets
20	Magnitue	Vector magnitude of a multi-feature representation (often derived)
21	Radius	Spatial radius from a center point (possibly PCA-based)
22	Covariance	Measure of variance shared between two variables (often in traffic pattern context)
23	Variance	Measure of data spread from the mean

By reducing the feature space from 46 to 23, the proposed selection method significantly improved computational efficiency, particularly in terms of training time and inference latency. However, it is important to note that this dimensionality reduction did not lead to a noticeable improvement in classification accuracy. Across both binary and multi-class settings, the overall accuracy remained largely unchanged, with only marginal variations observed in a few scenarios. Nevertheless, the reduced feature set enhanced the interpretability of the models by concentrating on the most relevant attributes, making the decision process more transparent and potentially more robust in real-world deployment.

4.6 Comparative Analysis with State-of-the-Art

To contextualize the performance of the proposed framework, a comparative analysis was conducted against recent state-of-the-art studies that employed the CIC-IoT2023 dataset. The results are outlined in Table 1, which summarizes reported accuracies and, where available, inference latencies.

In the binary classification setting, the proposed XGBoost model achieved 99.62% accuracy with a macro F1-score of 0.99 and a latency of 0.0004 milliseconds per sample. This performance is comparable to or better than the Transformer-based deep learning model reported by Tseng et al. [10], which achieved 99.57% accuracy but required specialized hardware and had a latency of approximately 5 µs/sample.

Federated learning approaches, such as the one proposed by Abbas et al. [3], reported 99.00% accuracy but lack inference time disclosures and incur communication overhead, making them less favorable for low-latency applications. Similarly, the LSTM-based model in [11] achieved 98.75% accuracy, yet sequential architectures typically entail higher latency and resource consumption.

In the multi-class setting, the proposed framework maintained strong performance across both the full 34-class task and the reduced 8-class taxonomy. On the 8-class classification, XGBoost reached 99.58% accuracy and a perfect weighted F1-score of 1.00, surpassing most previous work which typically reports accuracy in the 97–99% range without detailing latency or class-wise F1-scores.

Compared to traditional machine learning pipelines (e.g., Random Forest or Gradient Boosting with handcrafted features) [23], the present approach combines automatic feature selection with high-performance models trained on a reduced, yet expressive, feature set. Moreover, the use of LightGBM gain-based selection to reduce the dimensionality from 46 to 23 features contributed to lower inference time and improved model generalizability, a distinction often absent in related works.

Overall, this comparative analysis highlights that the proposed dual-pipeline architecture offers a favorable trade-off between accuracy, latency, and interpretability. Unlike many deep learning–based approaches that prioritize model complexity, our framework delivers competitive results using computationally efficient models that are well-suited for real-time intrusion detection in IoT networks.

5. Conclusion

This study introduced a dual-pipeline framework for intrusion detection in IoT networks, designed to address both binary and multi-class classification tasks using the CIC-IoT2023 dataset. The proposed system integrates LightGBM-based gain feature selection and a suite of high-performance classifiers, particularly gradient boosting ensembles such as XGBoost, CatBoost, and LightGBM.

Through extensive experimentation, the framework demonstrated high predictive performance across all classification scenarios. The binary classification pipeline achieved a peak accuracy of 99.62%, while the multi-class configurations both fine-grained (34 classes) and coarser (8 classes) attained accuracies of 99.45% and 99.58%, respectively. In all cases, models achieved sub-millisecond inference latency, confirming their suitability for real-time or resource-constrained environments.

Feature selection based on LightGBM gain values proved effective in reducing the dimensionality of the input space from 46 to 23 features. While this reduction did not lead to a significant change in classification accuracy, it substantially improved computational efficiency and enhanced model interpretability by emphasizing a core set of discriminative attributes primarily time-based and statistical flow features.

Importantly, the framework maintained reliable detection performance across both dominant and minority classes, addressing one of the key limitations in existing IoT intrusion detection systems. Misclassifications were rare and generally occurred between semantically similar attack categories, without compromising the system’s ability to separate benign from malicious traffic.

Future research could explore the integration of adaptive learning mechanisms to support concept drift in dynamic IoT environments. Additionally, combining interpretable boosting models with lightweight neural components may further improve resilience to novel or evolving threats, while maintaining real-time detection capabilities.

References

[1] J. Asharf, N. Moustafa, H. Khurshid, E. Debie, W. Haider, and A. Wahab, "A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions," *Electronics 2020, Vol. 9, Page 1177*, vol. 9, no. 7, p. 1177, Jul. 2020, doi: 10.3390/ELECTRONICS9071177.

[2] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," *Sensors 2023, Vol. 23, Page 5941*, vol. 23, no. 13, p. 5941, Jun. 2023, doi: 10.3390/S23135941.

[3] S. Abbas *et al.*, "A Novel Federated Edge Learning Approach for Detecting Cyberattacks in IoT Infrastructures," *IEEE Access*, vol. 11, pp. 112189–112198, 2023, doi: 10.1109/ACCESS.2023.3318866.

[4] Q. Li, Y. Liu, T. Niu, and X. Wang, "Improved Resnet Model Based on Positive Traffic Flow for IoT Anomalous Traffic Detection," *Electronics 2023, Vol. 12, Page 3830*, vol. 12, no. 18, p. 3830, Sep. 2023, doi: 10.3390/ELECTRONICS12183830.

[5] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Computer Networks*, vol. 76, pp. 146–164, Jan. 2015, doi: 10.1016/J.COMNET.2014.11.008.

[6] K. Jakotiya, V. Shirsath, and R. G. Mishra, "Feature Engineering Using Machine Learning Techniques on CIC-IOT-2023 Dataset," *Lecture Notes in Electrical Engineering*, vol. 1211 LNEE, pp. 717–727, 2025, doi: 10.1007/978-981-97-4359-9_64.

[7] S. M. Tseng, Y. Q. Wang, and Y. C. Wang, "Multi-Class Intrusion Detection Based on Transformer for IoT Networks Using CIC-IoT-2023 Dataset," *Future Internet 2024, Vol. 16, Page 284*, vol. 16, no. 8, p. 284, Aug. 2024, doi: 10.3390/FI16080284.

[8] M. M. Shtayat, M. K. Hasan, R. Sulaiman, S. Islam, and A. U. R. Khan, "An Explainable Ensemble Deep Learning Approach for Intrusion Detection in Industrial Internet of Things," *IEEE Access*, vol. 11, pp. 115047–115061, 2023, doi: 10.1109/ACCESS.2023.3323573.

[9] T. T. H. Le, R. W. Wardhani, D. S. Catur Putranto, U. Jo, and H. Kim, "Toward Enhanced Attack Detection and Explanation in Intrusion Detection System-Based IoT Environment Data," *IEEE Access*, vol. 11, pp. 131661–131676, 2023, doi: 10.1109/ACCESS.2023.3336678.

[10] O. Bello, S. Zeadally, and M. Badra, "Network layer inter-operation of Device-to-Device communication technologies in Internet of Things (IoT)," *Ad Hoc Networks*, vol. 57, pp. 52–62, Mar. 2017, doi: 10.1016/J.ADHOC.2016.06.010.

[11] Y. C. Wang, Y. C. Houng, H. X. Chen, and S. M. Tseng, "Network Anomaly Intrusion Detection Based on Deep Learning Approach," *Sensors 2023, Vol. 23, Page 2171*, vol. 23, no. 4, p. 2171, Feb. 2023, doi: 10.3390/S23042171.

[12] S. M. Tseng, Y. Q. Wang, and Y. C. Wang, "Multi-Class Intrusion Detection Based on Transformer for IoT Networks Using CIC-IoT-2023 Dataset," *Future Internet 2024, Vol. 16, Page 284*, vol. 16, no. 8, p. 284, Aug. 2024, doi: 10.3390/FI16080284.

[13] A. I. Jony and A. K. B. Arnob, "A long short-term memory based approach for detecting cyber attacks in IoT using CIC-IoT2023 dataset," *Journal of Edge Computing*, vol. 3, no. 1, pp. 28–42, Dec. 2023, doi: 10.55056/JEC.648.

[14] A. S. Jaradat, A. Nasayreh, Q. Al-Na'amneh, H. Gharaibeh, and R. E. Al Mamlook, "Genetic Optimization Techniques for Enhancing Web Attacks Classification in Machine Learning," in *2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, IEEE, Nov. 2023, pp. 0130–0136. doi: 10.1109/DASC/PiCom/CBDCom/Cy59711.2023.10361399.

[15] M. He, X. Wang, P. Wei, L. Yang, Y. Teng, and R. Lyu, "Reinforcement Learning Meets Network Intrusion Detection: A Transferable and Adaptable Framework for Anomaly Behavior Identification," *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 2477–2492, Apr. 2024, doi: 10.1109/TNSM.2024.3352586.

[16] A. Amouri, M. M. Al Rahhal, Y. Bazi, I. Butun, and I. Mahgoub, "Enhancing Intrusion Detection in IoT Environments: An Advanced Ensemble Approach Using Kolmogorov-Arnold Networks," *2024 International Symposium on Networks, Computers and Communications, ISNCC 2024*, Aug. 2024, doi: 10.1109/ISNCC62547.2024.10758956.

[17] K. S. Adewole, A. Jacobsson, and P. Davidsson, "Intrusion Detection Framework for Internet of Things with Rule Induction for Model Explanation," *Sensors 2025, Vol. 25, Page 1845*, vol. 25, no. 6, p. 1845, Mar. 2025, doi: 10.3390/S25061845.

[18] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans Knowl Data Eng*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.

[19] "Data Mining: Concepts and Techniques | ScienceDirect." Accessed: Jul. 11, 2025. [Online]. Available: https://www.sciencedirect.com/book/9780123814791/data-mining-concepts-and-techniques

[20] G. Ke *et al.*, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *Neural Information Processing Systems*, 2017.

[21] J. Bergstra, J. B. Ca, and Y. B. Ca, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, no. 10, pp. 281–305, 2012, Accessed: Jul. 11, 2025. [Online]. Available: http://jmlr.org/papers/v13/bergstra12a.html

[22] K. Hazelwood *et al.*, "Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective," *Proceedings - International Symposium on High-Performance Computer Architecture*, vol. 2018-February, pp. 620–629, Mar. 2018, doi: 10.1109/HPCA.2018.00059.

[23] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," *IEEE Trans Emerg Top Comput Intell*, vol. 2, no. 1, pp. 41–50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.