Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.076767

ARTICLE

A Lightweight Two-Stage Intrusion Detection Framework Optimized for Edge-Based IoT Environments

Chung-Wei Kuo[1,2,*] **and Cheng-Xuan Wu**[1]

[1]Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan
[2]Master's Program of Information and Communication Security, Feng Chia University, Taichung, Taiwan

*Corresponding Author: Chung-Wei Kuo. Email: cwkuo@fcu.edu.tw

Received: 26 November 2025; Accepted: 02 February 2026; Published: 09 April 2026

ABSTRACT: The rapid proliferation of the Internet of Things (IoT) has not only reshaped the digital ecosystem but also significantly widened the attack surface, leading to a surge in network traffic and diverse security threats. Deploying effective defense mechanisms in such environments is challenging, as conventional Intrusion Detection Systems (IDS) often struggle to balance computational efficiency with the reliable detection of low-frequency, high-impact threats, particularly within the tight resource constraints of edge devices. To address these limitations, we propose a lightweight, high-efficiency IDS framework specifically optimized for edge-based IoT applications, incorporating Mutual Information (MI)-based feature selection to reduce input dimensionality without compromising detection capabilities. The system employs a hierarchical two-stage classification strategy: Stage 1 utilizes a fast Decision Tree (DT) model to rapidly isolate and filter dominant Distributed Denial-of-Service (DDoS) traffic, thereby substantially reducing the computational burden for subsequent analysis. Subsequently, Stage 2 applies a soft-voting ensemble comprising Random Forest (RF), eXtreme Gradient Boosting (XGBoost), and Adaptive Boosting (AdaBoost) to accurately classify remaining non-DDoS traffic and address class imbalance. Extensive experiments on the large-scale CICIoT2023 dataset demonstrate that the proposed framework achieves a classification accuracy of 99.67%, while simultaneously reducing training and testing time by nearly 64.5% compared to traditional single-stage approaches. The model's robustness and generalization capabilities were further validated on the CICIoMT2024 and Edge-IIoTset datasets, representing medical and industrial scenarios, respectively, where it achieved an accuracy of over 94.7%. Empirical validation on a Cortex-A53-based SCADA testbed confirms the system's real-time practicality, with a complete training process executed in just 249 seconds. With an average inference latency of approximately 26.66 µs per flow and modest resource consumption, the framework satisfies strict industrial timing constraints, supporting its deployment for scalable, resource-aware threat detection in distributed IoT and edge-fog environments.

KEYWORDS: Internet of Things; intrusion detection systems; edge computing; soft voting; supervisory control and data acquisition

1 Introduction

The rapid expansion of Internet of Things (IoT) deployments across healthcare, transportation, and industrial automation has reshaped the digital ecosystem, but it has also widened the attack surface. Many IoT devices operate under tight resource constraints and insecure configurations, making them attractive targets for Distributed Denial-of-Service (DDoS) attacks, data exfiltration, and command-and-control exploitation [1,2]. Although machine-learning-based intrusion detection mechanisms (IDM) have advanced substantially, practical edge deployment remains challenging due to three persistent factors: (i) the

computational and memory demands of many IDS models exceed the capabilities of IoT endpoints and gateways; (ii) IoT security datasets are often highly imbalanced, hindering reliable detection of minority yet critical attacks; and (iii) rapidly evolving attack behaviors limit the effectiveness of static signature-based defenses, motivating adaptive learning-based approaches [3,4]. Consequently, edge-ready IDS solutions must balance detection robustness with resource-aware efficiency.

Recent benchmark datasets—most notably CICIoT2023—provide more realistic and diverse IoT traffic for evaluation [5,6]. Prior studies have reported strong performance using machine-learning classifiers on CICIoT2023, but these evaluations often rely on reduced subsets and offline or centralized settings [7,8]. Such settings do not fully satisfy three practical requirements for deployable edge IDS: (i) training and testing on the full-scale dataset distribution, (ii) systematic feature reduction to lower inference cost, and (iii) empirical validation on resource-constrained edge hardware.

To address this gap, we propose a lightweight, high-performance IDS framework explicitly aligned with recent research on resource-aware and deployable IoT/IoMT intrusion detection, where high-accuracy models often remain impractical in real-world settings due to limited edge resources, insufficient transparency, and limited consideration of edge–fog architectures [9]. Our contributions are threefold: (1) Full dataset usage—we train and validate on the complete CICIoT2023 dataset to better reflect realistic traffic distributions and improve generalizability across diverse attack classes [5,6]; (2) Mutual Information (MI)-based feature selection—we reduce input dimensionality to lower inference cost while preserving detection performance, thereby improving edge-side efficiency [3]; and (3) Edge-level validation—we deploy and evaluate the proposed IDS on a Raspberry Pi (Cortex-A53) platform to empirically demonstrate runtime feasibility under constrained resources, directly addressing deployability concerns emphasized in recent IoT/IoMT IDS surveys [10].

This study further differentiates itself from representative two-stage IDS approaches, including Kamal & Mashaly (2025) [4] and Zhang et al. (2023) [9], in terms of stage-wise design, dataset setting, and deployment validation. Our framework uses a binary Decision Tree (DT) in Stage 1 for fast DDoS filtering, followed by a Stage 2 soft-voting ensemble (Random Forest, XGBoost, and AdaBoost) for fine-grained classification of non-DDoS traffic. By contrast, Zhang et al. [9] apply LightGBM for Stage 1 normal/abnormal screening and a CNN-based classifier for Stage 2 detection, while Kamal & Mashaly [4] adopt deep-learning hybrid pipelines (e.g., Autoencoder–CNN and Transformer–DNN) with extensive resampling and outlier-handling strategies. Moreover, these studies are evaluated on datasets such as CSE-CIC-IDS2018 [9] or CICIDS2017/NF-BoT-IoT-v2 [4], whereas our work emphasizes full-scale CICIoT2023 training and direct edge-device validation to demonstrate real-time feasibility in realistic IoT settings. In addition, the DT-based Stage 1 design provides an interpretable screening mechanism (e.g., feature-importance evidence and rule-like decision logic), supporting transparency and operational trust at the edge—an increasingly important requirement for deployable IoT/IoMT IDS solutions. Finally, our modular two-stage architecture is naturally compatible with edge–fog partitioning (e.g., Stage 1 on-device and Stage 2 on near-edge/fog nodes) and can be extended in future work toward federated and privacy-preserving IDS learning paradigms, which are highlighted as open challenges in recent surveys.

The remainder of this paper is organized as follows. Section 2 reviews related work and the foundations of lightweight IDS design. Section 3 describes the proposed two-stage framework and implementation details. Section 4 presents experimental results and comparative evaluations. Section 5 concludes with implications and directions for future research.

2 Related Work

The inherent characteristics of IoT devices, namely their open interfaces and constrained hardware capabilities, render them susceptible to exploitation by cyber adversaries. To address this vulnerability, IDS solutions operate as proactive monitoring agents. These agents systematically analyze network and behavioral data streams to intercept potential threats before they escalate into actionable attacks.

Advancements in ML have greatly improved how well IDS can adapt and how smart it is, especially in IoT environments with limited computing power and memory. However, designing IDS architectures that simultaneously detect threats accurately and use minimal computing power remains a core technical challenge, especially when targeting deployment on edge-level or embedded IoT platforms. This section elaborates on the technical foundation of the proposed framework. We begin by reviewing core principles of intrusion detection in IoT settings. We then present the two-stage ML-based architecture developed in this study, detailing the algorithms employed at each stage and illustrating how their integration enhances detection effectiveness while maintaining compatibility with the stringent constraints of IoT devices.

2.1 Intrusion Detection Systems

In IoT environments, IDSs are commonly categorized as signature-based IDS (S-IDS) and anomaly-based IDS (A-IDS). S-IDS detects attacks using known signatures and typically provides high precision for previously seen threats, but it is less effective against zero-day or evolving attacks. In contrast, A-IDS models normal behavior and flags deviations, offering stronger generalization at the cost of potentially higher false positives [11,12].

To improve coverage and practicality in distributed IoT infrastructures, Otoum et al. proposed a hybrid IDS that combines S-IDS and A-IDS, achieving improved detection coverage with fewer false positives in fog-enabled IoT environments [13]. In this study, we adopt CICIoT2023, a large-scale IoT benchmark dataset developed by the Canadian Institute for Cybersecurity, which includes diverse attack families such as DDoS, DoS, scanning, web intrusion, brute force, spoofing, and Mirai botnets [14–16].

2.2 Stage 1 Machine Learning Algorithms

Because DDoS traffic often dominates IoT attack distributions, Stage 1 is designed as a binary screener that separates DDoS vs. non-DDoS flows. This hierarchical design reduces the computational burden and training complexity of the subsequent multi-class classifier. We evaluated four lightweight binary classifiers (Decision Tree, LDA, GNB, and Logistic Regression), as summarized in Table 1.

Table 1: Comparison of machine learning algorithms used in Stage 1.

Model	Characteristics
DT	A fast and interpretable model suitable for high-dimensional data. It segments the data space using hierarchical rules. However, DT is prone to overfitting and instability when facing noisy or imbalanced datasets.
LDA	A statistical method that reduces dimensionality while preserving class separability. It works well under the assumption of Gaussian distributions with equal covariances, which may limit its effectiveness on real-world nonlinear data.

(Continued)

Table 1 (continued)

Model	Characteristics
GNB	Known for its computational efficiency and suitability for small datasets, GNB assumes conditional independence between features. Despite its simplicity, this assumption often oversimplifies complex traffic patterns.
LR	A widely used binary classifier that models the probability of a class using a logistic function. While highly interpretable, LR is limited in capturing non-linear patterns.

Among the evaluated candidates, Decision Tree (DT) provided the best balance between accuracy, low training cost, and interpretability, making it suitable for edge-side real-time screening in IoT IDS deployments [17–21].

2.3 Stage 2 Machine Learning Algorithms

Stage 2 focuses on fine-grained classification of non-DDoS traffic, where several attack classes are relatively underrepresented but security-critical. To improve robustness and generalization [22], we adopt ensemble learning methods summarized in Table 2 [23]. Random Forest (RF) [24] enhances stability through bagging, whereas XGBoost [25,26] and AdaBoost [27] strengthen discrimination by emphasizing hard-to-classify instances (and incorporating regularization in XGBoost).

Table 2: Comparison of machine learning algorithms used in Stage 2.

Model	Characteristics
RF	RF aggregates predictions from multiple decision trees trained on bootstrapped data and feature subsets, enhancing stability, mitigating overfitting, and maintaining scalability.
XGBoost	A powerful boosting algorithm that builds trees sequentially by minimizing residual errors using gradient descent. XGBoost incorporates regularization and parallel processing, making it effective for complex, noisy data environments.
AdaBoost	AdaBoost focuses on correcting misclassifications by assigning higher weights to difficult instances in subsequent iterations. It is particularly useful in dealing with imbalanced datasets.

These classifiers are combined using a soft-voting strategy [28], where the final prediction is based on averaged class probabilities. Probability-level fusion is particularly helpful for imbalanced multi-class detection because it incorporates confidence information rather than relying only on majority votes.

Positioning vs. prior two-stage IDS studies. Our Stage 2 design differs from representative two-stage pipelines in both modeling choices and deployability considerations. Zhang et al. (2023) [9] employ LightGBM for Stage-1 normal/abnormal screening and a CNN for Stage 2 fine-grained classification on CSE-CIC-IDS2018, while Kamal & Mashaly (2025) [4] propose deep-learning hybrid pipelines (e.g., Autoencoder–CNN and Transformer–DNN) with extensive resampling/cleaning strategies (e.g., ADASYN-SMOTE/SMOTE and ENN) on CICIDS2017 and NF-BoT-IoT-v2. Beyond these two-stage frameworks, recent single-stage deep learning-based IDS studies also highlight domain-specific modeling for

IoT/IIoT/IoMT scenarios: Gueriani et al. (2025) [29] introduce BiGAT-ID, a hybrid transformer-based model integrating BiGRU, LSTM, and multi-head attention (MHA), designed to capture bidirectional temporal dependencies and enhance contextual feature representation for cross-domain security in medical and industrial IoT, achieving high accuracies (99.13% on CICIoMT2024 and 99.34% on EdgeIIoTset) with low inference latency; another work Akar et al. (2025) [30] proposes L2D2, an enhanced LSTM model tailored for multi-class intrusion detection in IoMT environments, which attains 98% accuracy across 19 attack classes on the CICIoMT2024 dataset; meanwhile, the Edge-IIoTset dataset study Ferrag et al. (2022) [31] validates the effectiveness of deep learning (alongside traditional machine learning) in both centralized and federated learning modes for IoT/IIoT intrusion detection, leveraging 61 high-correlation features extracted from network traffic, system logs, and alerts. In contrast to these deep learning-focused approaches—whether single-stage or two-stage—that prioritize accuracy through complex neural architectures, we adopt a resource-aware classical ensemble (RF + XGBoost + AdaBoost) with soft voting to maintain strong accuracy while keeping inference lightweight for edge deployment.

3 Proposed Method

3.1 System Architecture

Fig. 1 illustrates the overall architecture of the proposed IoT intrusion detection system, which is designed to maintain high detection accuracy with minimal computational overhead, thereby ensuring its suitability for deployment in resource-constrained IoT environments.

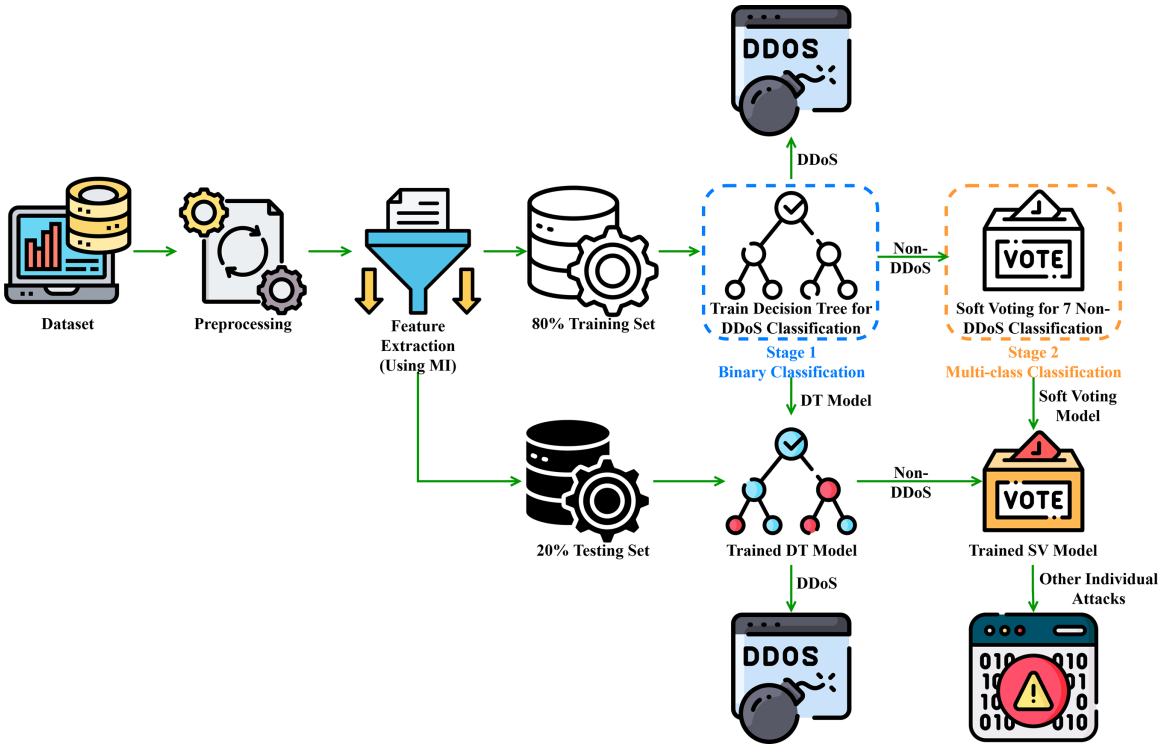

Figure 1: System architecture of the two-stage IoT intrusion detection framework.

The system is structured into five core functional modules, each of which serves a critical function within the detection pipeline:

1. Data Preprocessing: It cleans and normalizes the raw network traffic data to ensure the input is suitable for downstream analysis.
2. Feature Ranking and Selection: It uses statistical techniques to determine the most critical features, reducing the number of dimensions and the amount of computing power needed.
3. Two-Stage Classification Architecture: This module does a hierarchical classification by first figuring out if the traffic is DDoS, then looking at the rest of the traffic to see what kind of attack it is. It uses an ensemble learning strategy to integrate multiple classifiers, which enables it to detect attacks accurately while reducing computational complexity.
4. Decision Fusion: It combines the results of different classifiers using a "soft voting mechanism." This makes it more reliable and improves its accuracy.
5. Inference Output: It uses the fused predictions to produce the final detection result, which allows for real-time threat response.

Each module has been optimized for lightweight execution and carefully coordinated to maximize detection performance in dynamic IoT environments. The subsequent sections provide a detailed explanation of the role and operation of each module within the proposed system.

3.1.1 Data Preprocessing

This study employs the CICIoT2023 dataset, which contains network traffic from diverse IoT devices and encompasses multiple cyberattack behaviors. A comprehensive preprocessing pipeline was applied to ensure data quality and consistency, including handling missing values, filtering outliers, standardizing feature scales, and segmenting the data. Following established methodology, the dataset was split into training (80%) and testing (20%) subsets. This division enables objective, consistent model evaluation while preventing information leakage. All preprocessing operations that require estimating statistics (e.g., scaling parameters) are fitted on the training set and then applied to the test set to avoid leakage.

3.1.2 Feature Selection

To enhance classification efficiency and eliminate redundant or irrelevant features, this study uses the Mutual Information (MI) method for feature selection. By quantifying the dependency between each input feature and the target label, MI facilitates the identification of the most informative attributes. This process reduces computational overhead and improves the model's classification performance. A detailed explanation of MI calculation and implementation is provided in Section 3.2. MI scores are computed on the training set only, and the top-K ranked features are selected and applied consistently to the test set.

3.1.3 Binary Classification (Stage 1)

The first stage of the classification pipeline involves training a binary classifier to distinguish between DDoS and non-DDoS traffic. This step is designed to substantially reduce the computational load of the subsequent multi-class classification stage. Based on comparative evaluations, four candidate models were considered: DT, Linear Discriminant Analysis (LDA), GNB, and LR. After empirical testing, the model demonstrating the optimal balance between accuracy and computational efficiency was selected as the final Stage 1 classifier. The final DT configuration and hyperparameters are reported in Section 3.3 for reproducibility.

3.1.4 Final Decision Fusion

A soft voting mechanism is integrated into the final decision fusion phase to improve predictive consistency and alleviate the impact of class imbalance. Unlike hard voting, which determines the majority class based exclusively on discrete outputs, soft voting calculates the weighted average of class probabilities generated by each base classifier. The class with the highest aggregated probability is then selected as the final output. This probability-based approach enables the model to integrate nuanced confidence levels across classifiers, making more stable and accurate decisions. This advantage is particularly evident in scenarios where class boundaries are ambiguous or classes are underrepresented. Unless otherwise stated, equal weights are used for base learners, and class probabilities are aggregated by averaging.

3.1.5 Training and Evaluation Workflow

To improve reproducibility, Fig. 2 summarizes the complete training and evaluation workflow of the proposed two-stage IDS. First, CICIoT2023 traffic is preprocessed and split into training (80%) and testing (20%) subsets. All preprocessing statistics (e.g., scaling parameters) are fitted on the training set and then applied to the test set to prevent information leakage. Next, MI scores are computed using the training set only, and the selected top-K features are consistently applied to both training and testing data.

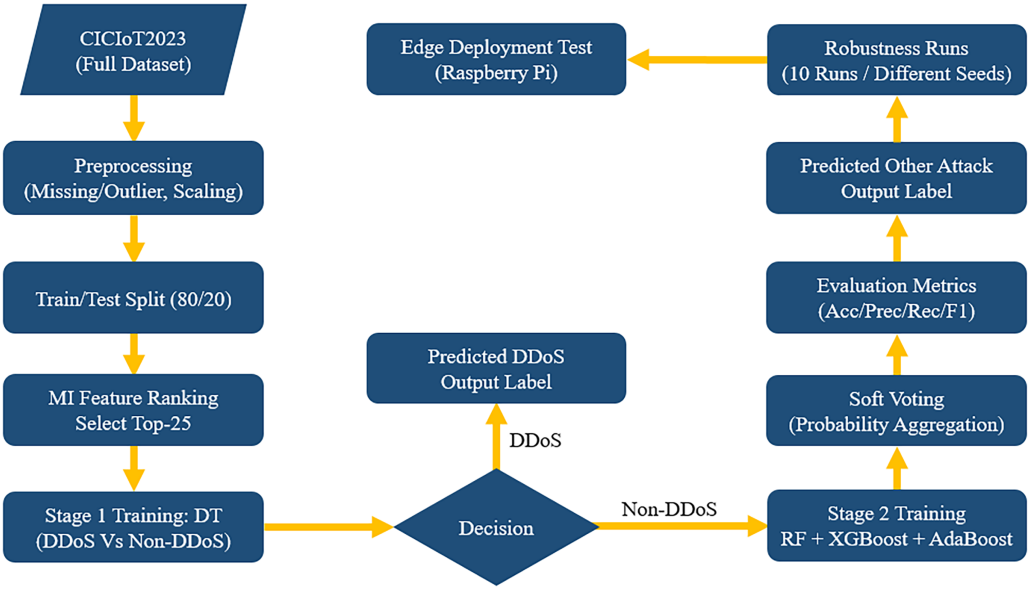

Figure 2: Complete training and evaluation workflow of the proposed two-stage IDS.

In Stage 1, a lightweight DT binary classifier is trained to distinguish DDoS from non-DDoS traffic, thereby reducing the computational burden of the subsequent multi-class stage. Samples predicted as non-DDoS are forwarded to Stage 2, where an ensemble of Random Forest, XGBoost, and AdaBoost is trained for fine-grained multi-class classification. The final decision is produced via probability-level soft voting by aggregating class probabilities across base learners.

Finally, the trained two-stage pipeline is evaluated using standard metrics (Accuracy, Precision, Recall, and F1-score). To verify robustness under the reported high performance, we repeat the complete training–evaluation process across multiple independent runs with different random seeds and report mean and dispersion statistics. In addition to offline evaluation, the trained models are deployed on an edge platform to validate runtime feasibility under constrained resources.

3.2 Feature Selection

The objective of feature selection is to identify the most significant features for model prediction from many potential features. This step plays a critical role in the machine learning pipeline. The objectives are to enhance model performance, mitigate the risk of overfitting, and reduce the time required for training and inference. Ultimately, this approach aims to improve the interpretability of the model.

The present study employs MI as a methodology for feature selection. MI is a filter-based feature selection technique that quantifies the correlation between features and the target variable. In evaluating the dependency between two random variables, the MI metric has been shown to reduce the uncertainty of one variable given the other. In the context of two random variables X and Y, the MI between X and Y can be calculated as follows:

$$I(X;Y) = \sum_{x\in X, y\in Y}^{p} (x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{1}$$

where:

- $p(x,y)$ denotes the joint probability distribution of X and Y.
- $p(x)$ and $p(y)$ represent the marginal probability distributions of X and Y, respectively.

MI is a valuable metric for quantifying the nonlinear relationship between features and target variables. This property confers upon it a notable advantage in analyzing complex data. Furthermore, this method is well-suited for processing continuous and discrete data, exhibiting great versatility. The present study utilizes the mutual_info_classif function, part of the scikit-learn library, to calculate the MI value of each feature and subsequently arrange the features in a hierarchical order.

3.3 Stage 1 Machine Learning Model (Binary Classification)

In the first stage, this study evaluated four machine learning algorithms for binary classification of DDoS attacks: DT, LDA, GNB, and LR. Each algorithm was assessed based on classification accuracy, computational efficiency, and suitability for high-volume DDoS traffic detection. Table 1 presents a comparative analysis of these models' performance.

- DT offers high interpretability and nonlinear adaptability but is prone to overfitting.
- LDA excels at data separability but assumes linear class boundaries.
- GNB suits high-dimensional data but relies on feature independence.
- LR is effective for linearly separable data but may struggle with complex patterns.

A key motivation for selecting DT as the Stage 1 screener is its intrinsic interpretability, which is valuable for IoT/edge IDS deployment where analysts require actionable reasons for alarms. DT decisions can be expressed as a sequence of human-readable feature-threshold rules (root-to-leaf path), enabling transparent auditing of why a flow is flagged as DDoS. To provide quantitative interpretability evidence, we report a feature importance visualization derived from the trained DT, as shown in Fig. 3. The figure highlights the most influential traffic features used by the Stage 1 classifier, indicating that the DT relies on a compact set of discriminative attributes for rapid DDoS screening. This interpretability not only improves practical usability but also supports edge deployment by maintaining a lightweight decision process.

The Stage 1 Decision Tree was implemented using Scikit-learn with default hyperparameters to maximize classification precision while maintaining the low-latency inference required for edge computing. The key configurations include:

- criterion ('gini'): Selected for its computational efficiency over entropy, reducing processing overhead for real-time detection.
- max_depth (None): Allows unrestricted tree growth to capture complex, high-dimensional attack patterns, directly contributing to the model's 99.99% accuracy.
- min_samples_split (2) & splitter ('best'): Ensures granular decision boundaries by rigorously evaluating all features to identify optimal split points at every node.

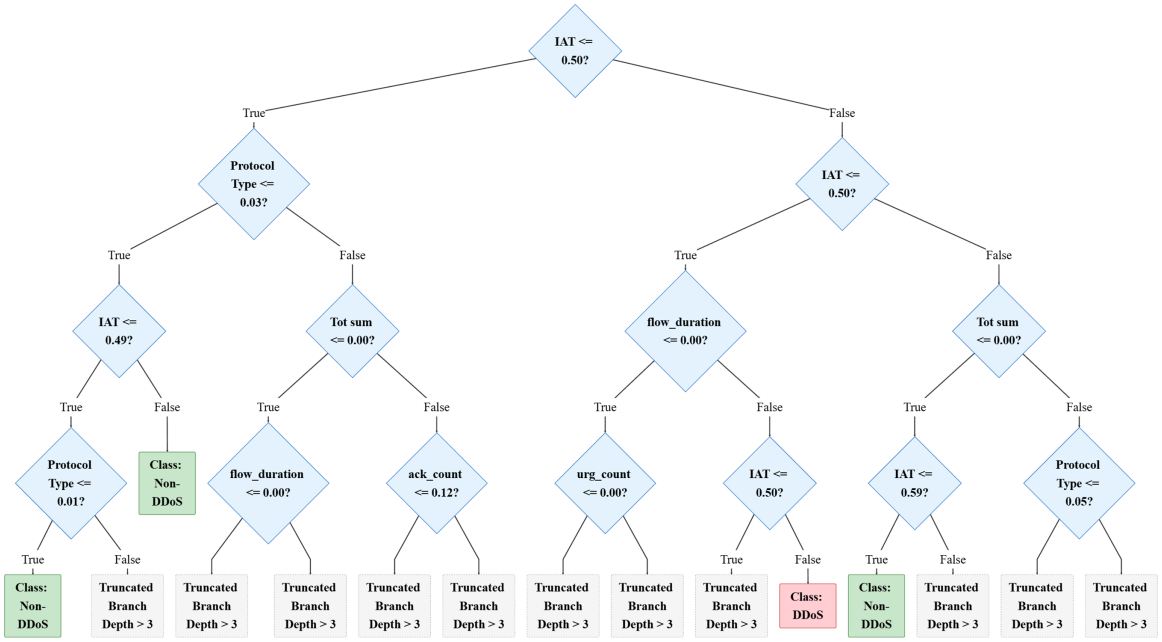

Figure 3: Visualization of the stage 1 decision tree logic (Top 3 Layers).

3.4 Stage 2 Machine Learning Model (Multi-Class Classification)

Following the successful elimination of DDoS attacks in the initial phase, the subsequent phase involves the detailed classification of the remaining seven non-DDoS attack types. To enhance classification efficacy, this study examined the following four ensemble learning methodologies:

- RF: The model formulates predictions using multiple decision trees, exhibiting high accuracy and low overfitting risk.
- XGBoost: Enhances model performance through gradient boosting, suitable for large-scale data.
- AdaBoost: Improves accuracy through weighted weak classifiers, and works well for difficult-to-classify samples.
- Extra Trees: Compared with RF, it can further improve training efficiency and reduce computational burden.

The following are experimental settings for each of the model classifiers we evaluated for stage 2:

1. RF Classifier: As the primary component of the soft-voting ensemble (weighted 2:1:1), the Random Forest model was deployed with Scikit-learn's default settings to ensure maximum stability and variance reduction. Key configurations include:
   - n_estimators (100): Utilizes 100 decision trees to construct a robust aggregate prediction. This ensemble size provides a strong balance between computational cost and the smoothing of errors from individual trees.

- max_depth (None): Similar to the Stage-1 model, fully grown trees are permitted. However, the bagging (bootstrap aggregating) mechanism inherent to RF effectively mitigates the overfitting risks usually associated with deep trees, ensuring high accuracy for the majority of classes.

2. XGBoost Classifier: The XGBoost model was utilized with standard defaults to leverage its scalable tree boosting system, which is particularly effective for capturing non-linear traffic patterns missed by traditional bagging methods. Key configurations include:
   - booster ('gbtree') & max_depth (6): Unlike the unconstrained depth of the RF model, XGBoost defaults to a depth of 6. This constraint acts as a regularization technique, preventing the model from memorizing noise and helping it generalize better to unseen minority attack signatures.
   - learning_rate (0.3): The default step size shrinkage prevents the model from adapting too quickly to specific data points, ensuring a steady convergence towards the optimal solution during the gradient descent process.
3. AdaBoost Classifier: AdaBoost was incorporated to specifically target hard-to-classify instances by sequentially correcting the errors of preceding weak learners. Key configurations include:
   - n_estimators (50): The default of 50 iterative estimators allows the model to focus on misclassified samples without incurring excessive training time.
   - algorithm ('SAMME.R'): This real boosting algorithm uses class probability estimates rather than discrete classifications. This is crucial for the Soft Voting mechanism, as it provides the granular probability distributions required to refine the final ensemble decision for underrepresented classes like Brute Force and Web-based attacks.
4. Extra Trees Classifier: The Extra Trees classifier was employed to further minimize model variance and computational overhead. Unlike Random Forest, this model adopts a more stochastic approach to tree building, implemented via Scikit-learn with default settings to enhance generalization. Key configurations include:
   - n_estimators (100): An ensemble of 100 decision trees is constructed to ensure robust predictive performance and stability across diverse traffic samples.
   - max_depth (None) & bootstrap (False): The model grows trees without depth restrictions using the entire dataset (no bootstrapping). This preserves the full information content of the training data, which is critical for identifying rare signatures in minority attack classes.
   - max_features ('sqrt') & Random Splitting: While it considers a subset of features (square root of the total) at each node, the defining characteristic is that split thresholds are drawn completely at random rather than searching for the optimal cut-point. This mechanism significantly reduces bias and training time compared to standard Random Forests.

As illustrated in Table 3, a performance comparison of the four models is presented. Ultimately, the soft voting mechanism was selected for decision fusion. The prediction probabilities of the model were combined through weighted averaging, as shown in Eq. (2).

$$\mathrm{P}\left(y = c|x\right) = \frac{1}{n}\sum_{i=1}^{n} p_i\left(c|x\right) \tag{2}$$

where:

- $p_i\left(c|x\right)$: the predicted probability that the $i - th$ classifier assigns to class c.
- n: the total number of classifiers participating in the voting process.

Table 3: Performance comparison of two-stage classifiers.

Model	Performance Description
RF	High efficiency and robust to overfitting through feature importance assessment; however, training time can be relatively long.
XGBoost	High accuracy and well-suited for large datasets with built-in regularization; however, hyperparameter tuning can be complex.
AdaBoost	Enhances weak learners and adapts well to complex samples; however, it can be sensitive to outliers.
Extra Trees	Fast training and suitable for high-dimensional features; however, it may yield lower accuracy than RF in some scenarios.

Unlike hard voting, soft voting employs weighted averaging to leverage the prediction probabilities from each model. This approach integrates multiple models' strengths, enhancing classification accuracy and robustness. The final predicted category is the category that corresponds to the maximum probability, as demonstrated in Eq. (3).

$$\hat{y} = \arg\max_{c} P\left(\left(y = c|x\right)\right) \tag{3}$$

This study employs a strategy of "detecting DDoS first and then classifying it," which offers the following advantages:

- Processing of imbalanced data. Distributed Denial of Service (DDoS) attacks represent a significant proportion of the CICIoT2023 statistics. If multi-category classification is performed directly, the classifier will be biased towards DDoS.
- It is essential to reduce the computational burden. Eliminating high-frequency attack categories first can reduce the number of samples and feature complexity that the subsequent classification model needs to process.
- Improving accuracy is essential for achieving optimal results. The two-stage strategy enables the first layer to concentrate on preliminary screening, while the second layer handles fine classification, aligning with the principles of divide-and-conquer. This strategy has proven stable and generalizable in numerous mathematical and experimental studies.

4 Experimental Results and Analysis

This section presents the experimental evaluation of the proposed two-stage IDS. The framework's performance is comprehensively assessed using the publicly available CICIoT2023 dataset, evaluating performance based on key metrics, including accuracy, precision, recall, F1-score, and inference efficiency.

4.1 Experimental Environment

The software and hardware configurations used in this study are summarized in Table 4. All experiments were conducted on an Ubuntu 22.04.3 LTS operating system, using Python 3.11.5 and widely adopted libraries for implementation and analysis.

Table 4: Specifications of the experimental environment.

Components	Specification
CPU/Memory	AMD Ryzen Threadripper 2950X 16 Core Processor/128 GB
Packages	Pandas, numpy, matplotlib, seaborn, sklearn, xgboost, joblib

4.2 Experimental Setup and Data Preprocessing

The CICIoT2023 dataset, developed by the Canadian Institute for Cybersecurity (CIC), was selected as the benchmark for this study. It provides realistic, large-scale IoT traffic collected from an experimental topology comprising 105 IoT devices, totaling approximately 46,686,579 records. The dataset encompasses 33 distinct attack scenarios organized into eight major categories: seven attack families and one normal traffic category. Available in both pcap and CSV formats, each record includes 47 extracted flow/window-based features, ensuring the dataset is rich in content and suitable for reproducible, real-world IDS validation. The distribution of attack types is detailed in Table 5.

Table 5: Attack categories in the CICIoT2023 dataset.

Class	Sub-Class
DDoS	ACK fragmentation, UDP flood, SlowLoris, ICMP flood, RSTFIN flood, PSHACK flood, HTTP flood, UDP fragmentation, TCP flood, SYN flood, SynonymousIP flood, ICMP Fragmentation
DoS	TCP flood, HTTP flood, SYN flood, UDP flood
Recon	Ping sweep, OS scan, Vulnerability scan, Port scan, Host discovery
Web-Based	SQL injection, Command injection, Backdoor malware, Uploading attack, XSS, Browser hijacking
Brute Force	Dictionary brute force
Spoofing	Arp spoofing, DNS spoofing
Mirai	GREIP flood, Greeth flood, UDPPlain
Normal	Normal

To improve clarity, we group the labels in Table 5 by attack family (high-level category) and list the corresponding sub-classes under each family, following the dataset's original taxonomy. The seven attack families are summarized below:

1. Availability attacks (volumetric/service disruption).
   - DDoS: distributed flooding from multiple compromised devices to exhaust network/service resources.
   - DoS: single/few-source resource exhaustion targeting service availability.
   - Mirai botnet: IoT malware that compromises devices (often via weak/default credentials) and subsequently orchestrates large-scale attacks, frequently including DDoS campaigns.
2. Probing and information gathering.
   - Reconnaissance (Recon): active/passive scanning to discover network configuration, open ports, services, and potential vulnerabilities prior to exploitation.

3. Application/credential-driven compromise.
    - Web-based attacks: exploitation of web application weaknesses to steal data, alter behavior, or degrade service reliability/availability.
    - Brute Force attacks: systematic password guessing (dictionary and exhaustive search) to obtain unauthorized access.
4. Deception and identity manipulation.
    - Spoofing: falsifying identifiers (e.g., IP/MAC/source fields) to bypass defenses or impersonate legitimate entities.

4.3 Experimental Results

4.3.1 Data Preprocessing

The CICIoT2023 dataset was first standardized to ensure consistent feature scaling across all input variables. The original dataset contains 34 distinct class labels, which were reclassified into seven attack categories plus one normal traffic class based on the taxonomy provided in Table 6, where the class distribution and sample counts across these eight categories are also summarized.

Table 6: Traffic class distribution.

Label	Class Distribution	Train	Test
DDoS	72.686%	27,189,129	6,795,431
DoS	17.387%	6,472,295	1,618,443
Mirai	5.720%	2,106,017	528,107
Normal	2.318%	878,851	219,344
Spoofing	1.061%	388,993	97,511
Recon	0.748%	283,692	70,873
Web-Based	0.053%	19,837	4992
BruteForce	0.023%	10,449	2615

4.3.2 Dataset Imbalance Analysis

The class distribution analysis of the CICIoT2023 dataset, as shown in Table 6, shows a big difference in traffic categories. DDoS attacks are the most common type, accounting for about 73% of all samples. Brute force and web-based attacks are less common, making up less than 0.1% each. Shaped in this way, distributions can strongly influence the training of models. This can lead to poor recall for attack types that are not represented, which are critical for IoT security.

To deal with this problem, the suggested plan has two parts. First, Stage 1 uses a binary classification approach. This approach separates DDoS traffic from non-DDoS flows. This reduces the class imbalance in the analysis. This process breaks down the problem into two simpler steps, which makes it easier to detect the majority class while keeping the other classes in check. Second, Stage 2 uses a soft voting ensemble that combines RF, XGBoost, and AdaBoost classifiers. Overall, these design choices reduce the negative impact of dataset imbalance by dividing the classification tasks and using ensemble-based learning to improve general performance. This allows the framework to reliably detect both high-frequency and minority attack types. This approach makes it useful in the real world, where problems with class imbalance are common.

4.3.3 Feature Ranking

MI was used to evaluate the relevance of each feature concerning the target variable, serving as a key criterion for feature selection. Based on the MI scores computed using Eq. (1), features were ranked in descending order of informativeness. The results showed that the top 23 features contributed significantly to model accuracy. In contrast, Features ranked 31st and below showed negative or near-zero MI values, indicating negligible additional predictive value.

To determine the optimal feature subset, we focused on fine-tuning the selection range between features ranked 23rd to 31st, as these exhibited borderline MI scores that might offer marginal improvements in accuracy. The most effective feature combination was identified through iterative testing and comparison to balance training efficiency and classification performance. The complete ranking results are presented in Table 7.

Table 7: Feature Ranking Based on MI.

Sort	Feature	MI	Sort	Feature	MI
1	IAT	0.853210	**17**	rst_count	0.189326
2	Totsum	0.570154	**18**	urg_count	0.178389
3	Magnitude	0.538441	**19**	syn_count	0.119756
4	Min	0.521251	**20**	HTTPS	0.059965
5	Totsize	0.494339	**21**	ack_count	0.033488
6	AVG	0.469948	**22**	ack_flag_number	0.031644
7	Max	0.462722	**23**	fin_count	0.024392
8	Header_Length	0.379621	**24**	SSH	0.000339
9	Rate	0.242756	**25**	DNS	0.000297
10	Srate	0.242667	**26**	ARP	0.000171
11	Std	0.225085	**27**	cwr_flag_number	0.000170
12	Radius	0.223886	**28**	IRC	0.000122
13	Covariance	0.220497	**29**	DHCP	0.000053
14	flow_duration	0.202604	**30**	ece_flag_number	0.000036
15	Variance	0.194245	**31**	Telnet	0.000027
16	ProtocolType	0.194153			

4.3.4 Evaluation Metrics

To comprehensively evaluate the performance of the proposed model, we employed standard classification metrics, including Accuracy, Precision, Recall, and F1-Score. In addition, a confusion matrix was used to analyze the classification outcomes further.

The confusion matrix consists of the following four elements:

- True Positive (TP): Correctly classified positive samples.
- False Positive (FP): Negative samples incorrectly classified as positive.
- True Negative (TN): Correctly classified negative samples.
- False Negative (FN): Positive samples incorrectly classified as negative.

The definitions and mathematical formulations of the evaluation metrics are presented in Eqs. (4)–(7).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$\text{F1} - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

4.3.5 Stage 1 Classification Model

Fig. 4 compares how well four machine learning algorithms worked for Stage 1 binary classification. DT, Naïve Bayes, LR, and LDA. DT performed best with an accuracy of 99.99%, much better than the other models (Naïve Bayes: 81.52%, LR: 83.86%, LDA: 83.42%). This excellent result shows that DT can easily tell the difference between DDoS and non-DDoS traffic. This is important for reducing the computational workload, especially in environments that don't have a lot of resources. The classification accuracy is almost perfect, which shows that DT can accurately identify DDoS attack patterns while correctly identifying harmless traffic. This supports the framework's design goal of creating a lightweight yet highly accurate filter in the early stages.

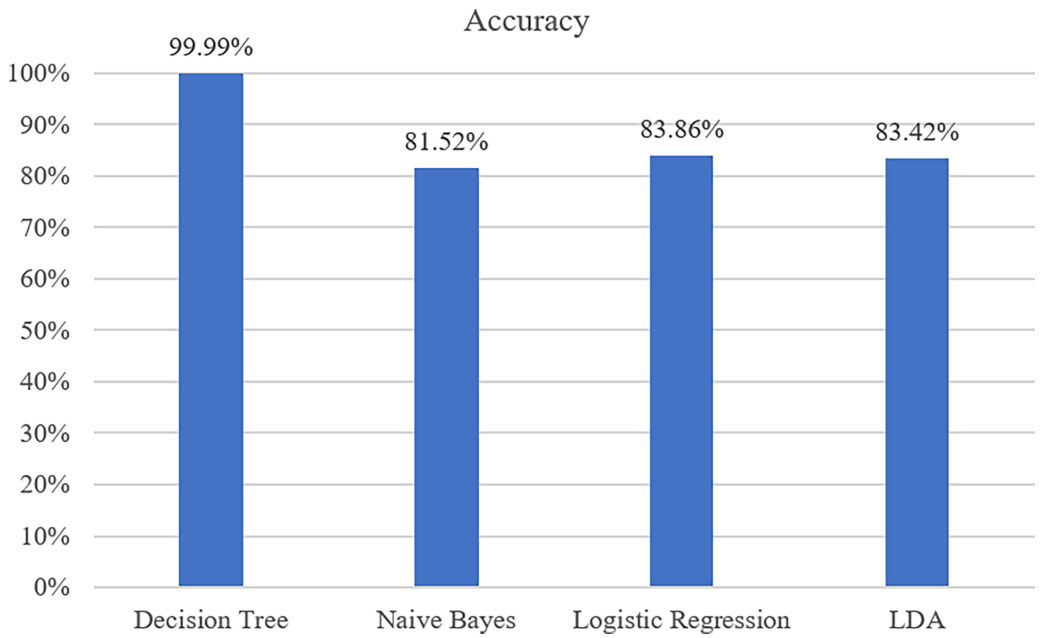

Figure 4: Performance comparison of multi-class classification algorithms for non-DDoS traffic.

4.3.6 Stage 2 Classification Model

Fig. 5 compares how well four different classifiers worked in Stage 2 to identify different types of attacks and normal traffic. The evaluated models include XGBoost, AdaBoost, Extra Trees, and RF. The results show that RF had the highest accuracy at 98.77%, with XGBoost close behind at 98.62% and Extra Trees at 98.53%.

AdaBoost had the lowest accuracy at 94.97%. These findings show that ensemble methods that use bagging and boosting are better at dealing with the complexity and class imbalance that are part of IoT network traffic. These results show that Stage 2 of the proposed framework effectively deals with the challenges of detecting intrusions in more than one category. It makes sure that attacks are well-defined and organized, so they can be stopped quickly.

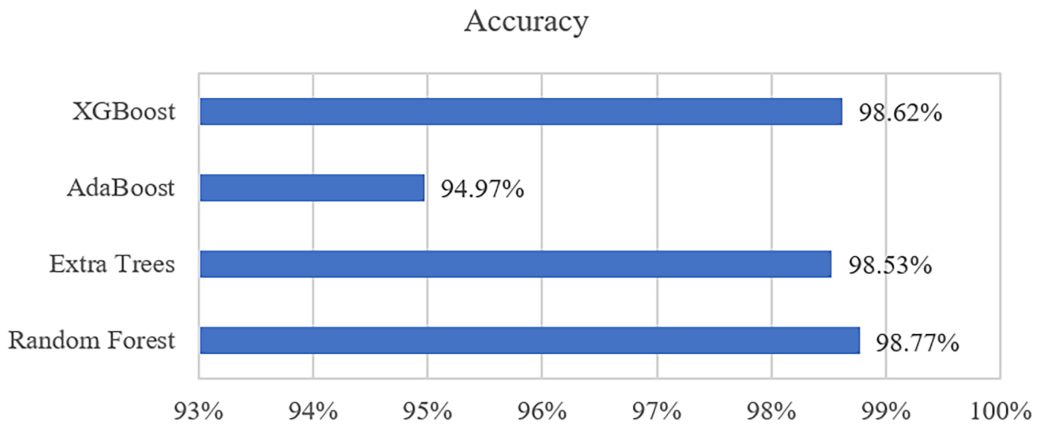

Figure 5: Performance comparison of different machine learning algorithms in Stage 2.

As outlined in Section 3.4, the soft voting strategy aggregates class probabilities from each model. Specifically, each classifier outputs a predicted probability $p_i(c|x)$ for sample x belonging to class c. The predicted class $\hat{y}$ is then selected using Eq. (3) by choosing the class with the maximum probability. As shown in Table 8, RF exhibited the highest standalone accuracy for multi-class (non-DDoS) classification. To compensate for the comparatively weaker performance of other models in detecting less frequent attack types, such as Web-based, Spoofing, and Brute Force, RF was assigned a higher voting weight in the ensemble. This strategy significantly reduced misclassifications while maintaining overall high S and F1-scores.

Table 8: Performance evaluation of soft voting combinations.

Voting Combination	Accuracy	Precision	Recall	F1-Score
		(%)		
2 × RandomForest + ExtraTrees + AdaBoost	98.7571	98.7649	98.7571	98.7218
2 × RandomForest + ExtraTrees + XGBoost	98.7839	98.794	98.7839	98.7483
2 × RandomForest + AdaBoost + XGBoost	98.7924	98.7999	98.7924	98.7571

Based on the soft voting mechanism (Eqs. (2) and (3)), the final ensemble achieved over 98.7% accuracy across all attack classes and effectively reduced false positives and false negatives. These results highlight the practical value of the proposed hierarchical IDS framework in securing IoT environments.

4.3.7 Integrated Results

After establishing the optimal voting ensemble, we further evaluated its performance using various feature subset sizes ranked by MI. The goal was to identify a configuration that maintains high predictive accuracy while reducing computational load and eliminating feature redundancy. Specifically, MI-ranked

features from positions 23 to 31 were selected for evaluation. Several soft voting models were trained and tested under different feature count configurations, with results presented in Table 9.

Table 9: Model performance under varying feature counts.

Feature Number	Accuracy	Precision	Recall	F1-Score
	(%)			
23	98.8082	98.8142	98.8082	98.7778
24	98.8028	98.8088	98.8028	98.7715
25	98.8296	98.8354	98.8296	98.8020
26	98.8181	98.8240	98.8181	98.7891
27	98.8171	98.8231	98.8171	98.7883
28	98.8114	98.8177	98.8114	98.7819
29	98.8046	98.8108	98.8046	98.7737
30	98.8007	98.8068	98.8007	98.7689
31	98.7957	98.8019	98.7957	98.7635

The experiment results show that choosing the top 25 features based on MI gets the best and most reliable results for all the ways to measure success, like accuracy, precision, recall, and F1-score. This setup balances how well the system works with how much it uses in terms of computing power by eliminating extra features and reducing the chance of performance issues.

Accordingly, the final model adopts these top 25 MI-ranked features as the input set for training the proposed two-stage classification framework. This architecture employs a DT classifier in the first stage to detect and filter DDoS traffic. The remaining non-DDoS traffic is then passed to the second stage, where a weighted soft voting ensemble is used for multi-class classification. This ensemble integrates three classifiers: RF, AdaBoost, and XGBoost. To reflect the superior standalone performance of the RF classifier observed in earlier experiments, the voting weights are assigned in a ratio of 2:1:1, respectively. This weighting enhances the overall detection accuracy and contributes to improved model stability.

Fig. 6 presents the confusion matrix of the final classification results on the CICIoT2023 dataset, illustrating the relationship between predicted and actual labels across nine traffic categories: DDoS, DoS, Mirai, Normal, Spoofing, Reconnaissance, Web-based, and Brute Force attacks. The strong diagonal dominance observed in the matrix indicates high true positive rates for most classes. Notably, the model successfully identified over 6.79 million DDoS samples and 1.62 million DoS attacks, reflecting strong detection capability for majority-class attacks.

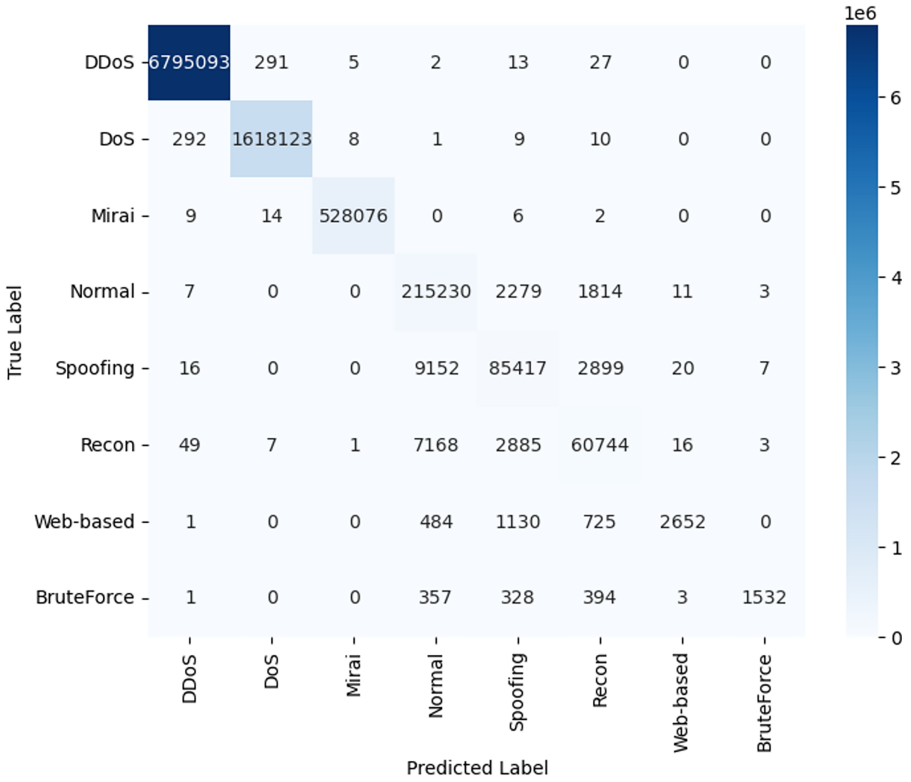

Figure 6: Confusion matrix of the final classification results on the CICIoT2023 dataset.

Although the overall confusion matrix indicates strong performance, minority classes—particularly Web-based and BruteForce attacks—remain more challenging due to limited samples and overlapping traffic characteristics with other non-DDoS categories. In our baseline setting (without resampling), the precision for Web-based and BruteForce attacks is 53.125% and 58.585%, respectively, suggesting that a portion of predictions for these rare classes are confused with visually similar behaviors in the feature space. To further investigate whether class-imbalance mitigation can improve detection completeness, we conducted an additional experiment using SMOTE oversampling to increase each minority class to 28,000 samples. This intervention substantially improved recall to 71.23% (Web-based) and 78.56% (BruteForce), indicating improved sensitivity to rare attacks. However, the accuracy dropped to below 60%, revealing a clear precision–recall trade-off: oversampling increases true positive coverage but also increases false positives for these rare classes.

Given that our two-stage design already improves minority-class discrimination through stage-wise decomposition and soft-voting aggregation in Stage 2, we prioritize maintaining precision to reduce false alarms in operational settings. Therefore, the final results are obtained without minority-class oversampling, while the SMOTE experiment is included as an ablation study to demonstrate the trade-offs and potential future directions for recall-oriented optimization.

The relatively lower detection performance for Brute Force and Web-based attacks is primarily driven by the severe class imbalance within the CICIoT2023 dataset, where these categories represent less than 0.1% of the total samples compared to the overwhelming dominance of DDoS traffic. Furthermore, the proposed model relies on flow-based statistical features (e.g., IAT, Rate) selected via Mutual Information, which are highly effective for identifying volumetric anomalies but lack the application-layer granularity required to

distinguish specific payloads. Consequently, these "silent" attacks exhibit traffic patterns that significantly overlap with legitimate behavior, leading to a higher rate of misclassification with Normal traffic compared to the distinct, high-velocity profile of DDoS attacks.

Furthermore, attacks on minority classes, such as Brute Force and web-based intrusions, were classified with reasonable accuracy, despite the small number of samples available. These results show that the model can generalize well without overfitting. It maintains high precision and recall across different types of attacks in a large, real-world IoT dataset.

In addition, a comparative analysis with recent studies that utilized the full CICIoT2023 dataset [5,11,15,16] confirms the superiority of the proposed framework. As summarized in Table 10, the two-stage IDS consistently outperforms existing approaches in all key evaluation metrics, validating its effectiveness in delivering high accuracy and computational efficiency. Furthermore, a comparison with recent studies using deep learning methods [29–31] further confirms that all indicators of this study are higher than those of other research results. The deep learning-based approaches and their corresponding performance metrics compared with our framework are presented in Table 11.

Table 10: Performance comparison with previous studies.

Reference	Accuracy	Precision	Recall	F1-Score
		(%)		
Our Method	99.67	99.67	99.67	99.66
Kumar et al. [5]	99.95	98.13	99.97	99.04
Gheni et al. [11]	97.46	97.00	97.00	97.00
Tseng et al. [15]	99.46	99.48	99.45	99.47
Neto et al. [16]	98.90	86.32	89.04	87.63

Table 11: Performance comparison with deep learning previous studies.

Reference	Accuracy	Precision	Recall	F1-Score
		(%)		
Our Method	99.67	99.67	99.67	99.66
[29]	99.13	99.13	99.13	99.13
[30]	98.00	98.00	98.00	98.00
[31]	96.56	94.59	91.28	92.62

4.3.8 ROC Curve and AUC Analysis

To further evaluate the generalization performance of the proposed two-stage IDS framework and examine potential overfitting issues, ROC (Receiver Operating Characteristic) curves and AUC (Area Under the Curve) values were plotted for the CICIoT2023 dataset, as shown in Fig. 7. These metrics provide an intuitive assessment of the model's discriminative ability across different attack classes, independent of classification thresholds.

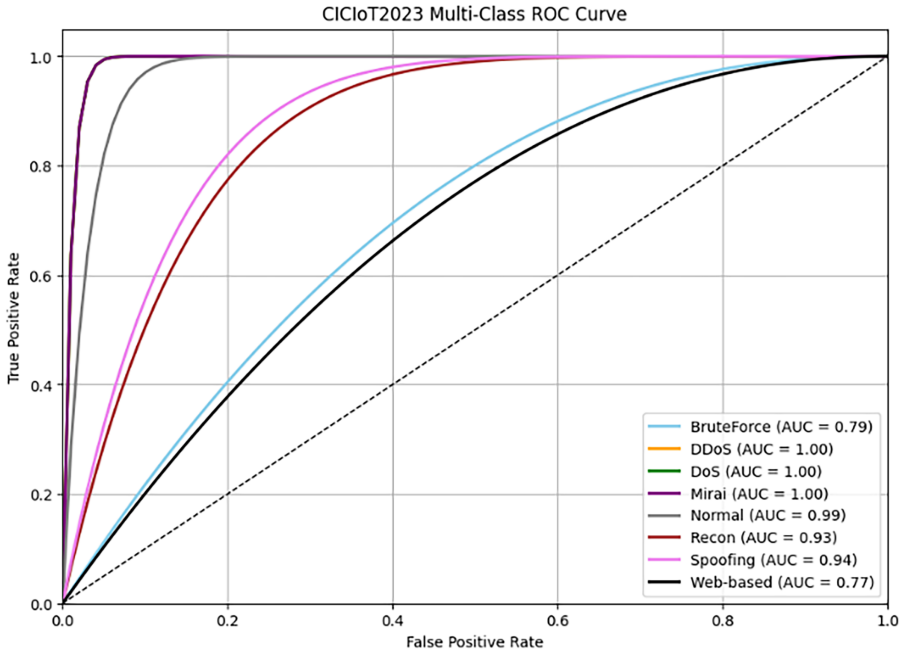

Figure 7: ROC curve for the proposed two-stage IDS model on the CICIoT2023 dataset.

For the CICIoT2023 dataset, the results show that the proposed model gets almost perfect scores for critical classes like DDoS (AUC = 1.00), DoS (AUC = 1.00), and Mirai (AUC = 1.00). This shows that it can detect problems well while identifying a few false positives. Meanwhile, attacks that target minority classes, like web-based attacks and brute force, have lower AUC values. These results suggest that the model works well for common and high-volume attack types, but more data or targeted sampling might be needed to improve detection of underrepresented classes.

Overall, the high AUC scores across different classes and datasets show that the model performs well and is not likely to be overfit. This is also supported by the small difference between the training and testing accuracy results. This suggests that the framework performs well without relying too much on certain training data patterns.

4.4 Method Evaluation

In the CICIoT2023 dataset, DDoS attacks represent the most frequent and voluminous category of malicious traffic. We implemented a two-stage IDS framework to alleviate computational burdens while maintaining high detection performance. Stage 1 employs a lightweight decision tree classifier in this architecture to isolate DDoS traffic from all other categories. Subsequently, Stage 2 applies a soft-voting ensemble classifier, comprising RF, XGBoost, and AdaBoost, to further classify the remaining non-DDoS traffic into its respective attack types. This staged design balances efficiency and accuracy, making it a promising candidate for real-time deployment in resource-constrained IoT edge environments.

To demonstrate the effectiveness of this design, we conducted a comparative evaluation against a single-stage soft-voting model that performs direct multi-class classification. The results are summarized in Table 12.

Table 12: Two-stage model evaluation on CICIoT2023 dataset.

Metric\Method	One-Stage	Two-Stages
Accuracy	99.671%	99.674%
Precision	99.672%	99.675%
Recall	99.671%	99.674%
F1-Score	99.662%	99.666%
TrainTime	5 h 6 min 52 s	1 h 48 min 55 s
TestTime	1 min 52 s	33 s

Statistical validation and robustness. Given the very high reported performance, we further assessed robustness by repeating the complete two-stage training–evaluation pipeline 10 independent runs with different random seeds. Across runs, the variation of key metrics (Accuracy, Precision, Recall, and F1-score) remained within 0.2%, indicating stable performance and reducing the likelihood that the results are driven by a single favorable split or initialization. We report the mean ± standard deviation (and 95% confidence intervals, where applicable) over the 10 runs in Table 13.

Table 13: Mean, Std, 95% CI, and range across 10 runs.

Metric	Mean	Std	95% CI (±)	Min	Max	Range (Max–Min)
Accuracy	99.6644%	0.0705	0.0437	99.5647%	99.7821%	0.2174
Precision	99.6690%	0.0812	0.0503	99.5610%	99.7785%	0.2175
Recall	99.6928%	0.0630	0.0390	99.5860%	99.7820%	0.1960
F1-Score	99.6584%	0.0498	0.0309	99.5952%	99.7471%	0.1519

To further assess the cross-domain generalization of the proposed lightweight two-stage IDS, we conducted additional evaluations on two widely adopted IoT security benchmarks, CICIoMT2024 and Edge-IIoTset, which represent medical IoT and industrial IoT scenarios, respectively. CICIoMT2024 emulates modern IoMT deployments where MQTT-based communication is prevalent, and the threat landscape includes DDoS, spoofing, MQTT-targeted exploits, and reconnaissance behaviors. In contrast, Edge-IIoTset reflects industrial IoT monitoring conditions with diverse attack families, such as credential-based intrusion attempts, injection attacks, and malware-related activities, resulting in traffic patterns with distinct operational periodicity and protocol heterogeneity. These differences provide a rigorous test of robustness beyond the original CICIoT2023 environment.

As reported in Table 14, under the original hyperparameter configuration defined in Sections 3.3 and 3.4, the proposed framework demonstrates strong transferability, achieving 99.80% accuracy on CICIoMT2024 and 94.73% accuracy on Edge-IIoTset. Importantly, all corresponding evaluation metrics remain consistently high, indicating that the hierarchical design and feature-efficient modeling strategy can preserve reliable detection performance even when the traffic distributions and attack characteristics shift across domains.

Table 14: Generalization performance evaluation results.

	Metric	Original	With SMOTE
		(%)	
CICIoMT2024	Accuracy	99.80	99.39
	Precision	99.82	99.45
	Recall	99.80	99.39
	F1-Score	99.80	99.41
EdgeIIoTset	Accuracy	94.73	94.65
	Precision	94.82	94.76
	Recall	94.73	94.65
	F1-Score	94.74	94.66

To understand the effect of class imbalance better, SMOTE-based over-sampling was performed on the minority level categories to populate these classes to approximately 20,000 instances each. This allowed for model retraining. The results indicate that when balancing the classification performance remains stable. The accuracy and recall of the model on CICIoMT 2024 decreased slightly (from 99.80% to 99.39%), while there was virtually no difference in the F1-score (99.80% vs. 99.41%). When calculated on all metrics at Edge IIoT Set, it was observed that accuracy decreased by a small amount from 94.73% to 94.65%, precision decreased from 94.82% to 94.76%, and the F1-score decreased from 94.74% to 94.66%. A consistent range of changes in all metrics indicates that the effect of sampling alone does not have a major impact on overall classification performance since the previously proposed model already incorporates the main discriminative characteristics of the data and extracts compact features from the data via two stage decomposition techniques.

The findings shown in Table 14 indicate how well the proposed lightweight IDS performs across a wide range of heterogeneous IoT domains, with stable and generalised performance from all instances of both IoT domain variations. By maintaining high-quality detection regardless of many differences in dataset characteristics, the framework has demonstrated practical value for real-world deployment within medical, industrial and general-purpose IoT domains without needing to be re-architected or require extreme amounts of hyperparameter tuning.

Beyond the performance comparison, it is important to clarify the practical differences, advantages, and trade-offs between single-stage and two-stage IDS designs. A single-stage multi-class classifier learns a direct mapping from input features to all attack labels in one model. In contrast, a two-stage IDS decomposes the task into a coarse screening step followed by fine-grained classification, allowing Stage 1 to quickly filter the dominant category (DDoS) using an interpretable and lightweight model, while Stage 2 focuses on the remaining non-DDoS traffic where finer distinctions are required.

From an efficiency perspective, the two-stage design reduces average inference cost because a large fraction of traffic can be decided by Stage 1 with minimal computation, which is particularly beneficial under edge constraints. It also improves scalability by enabling different deployment strategies (e.g., Stage 1 at the edge for immediate filtering and Stage 2 at the edge or near-edge, depending on resources). However, two-stage designs introduce potential error propagation: if Stage 1 misroutes samples (e.g., non-DDoS incorrectly flagged as DDoS or *vice versa*), Stage 2 may not have the opportunity to correct those errors. Therefore, the Stage-1 classifier must prioritize robust screening performance, and the overall system should be evaluated not only by accuracy but also by per-class recall and misclassification patterns. Despite this trade-off, our

results show that the two-stage framework provides a favorable balance between real-time feasibility and strong detection performance for large-scale IoT traffic.

For the CICIoT2023 dataset, the single-stage model required over 5 h of training time, whereas the combined training time of both classifiers in the proposed two-stage architecture was less than 2 h. During inference, the single-stage model took approximately 2 min to process the test set, while the two-stage system completed the same task in about 30 s. This represents a 3× reduction in inference time, offering a substantial runtime advantage.

In terms of practicality, these improvements significantly reduce training and operational overheads, particularly beneficial for large-scale IoT deployments, where real-time detection and rapid model updates are essential. To further assess the real-world applicability of the model, we deployed the trained two-stage IDS on both general-purpose computers and low-power edge computing platforms. Specifically, we evaluated performance on a Raspberry Pi 3, which features a Cortex-A53 CPU and typifies hardware used in SCADA or embedded control systems.

To comprehensively evaluate the real-time feasibility of the proposed lightweight two-stage IDS on edge hardware, we performed extensive experiments on a Raspberry Pi 3. For each traffic category, the system processed 10,000 inference instances while recording CPU utilization, memory consumption, and inference latency, as summarized in Table 15. The proposed IDS demonstrates strong operational efficiency under resource constraints. Batch inference times are consistently under 2 s across the board (DDoS = 0.108 s) through Benign (1.902 s). Average batch inference time was 1.19 s. Average instance latency was approximately 119 μs for every prediction. Peak CPU usage during batch inference did not exceed 35.78%. Maximum memory used per instance prediction did not exceed 8.45 MB confirming that this framework is a very lightweight and therefore well suited for IoT security edge computing.

Table 15: Average real-time operational metrics of the two-stage IDS on raspberry Pi 3.

Label	CPU Usage (%)	Memory Consumes (MB)	Inference Time (s)
DDoS	18.36	8.45	0.108
DoS	28.52	5.02	1.093
Mirai	23.3	1.62	1.050
Benign	21.94	0.912	1.902
Recon	26.54	0.336	1.688
Spoofing	35.78	0.112	1.684
Web-based	29.43	0.256	0.873
Bruteforce	23.66	1.18	1.137

To further contextualize the real-time capability under industrial requirements, IEC 61850-5 specifies stringent time performance constraints for time critical substation automation messages. In particular, trip messages (Type 1A) are typically expected to achieve transmission times on the order of 3 ms, while other fast event driven messages are commonly constrained within 10 ms to 20 ms depending on the performance class. Compared with these requirements, the proposed IDS achieves sub-millisecond inference latency per instance on a low-power Raspberry Pi platform, indicating that it can satisfy the real-time responsiveness demanded by mission critical smart grid and industrial control deployments while maintaining minimal resource overhead.

4.5 Deployment Feasibility and Practical Performance Analysis

In this section, we transition from theoretical metric evaluation to a practical assessment of deployment feasibility. By implementing the framework on resource-constrained edge computing platforms, we validate the real-world viability of our proposed two-stage architecture within ecosystems.

4.5.1 Resource-Aware Performance on Heterogeneous Hardware

To evaluate cross-platform stability, benchmarks were conducted across a range of hardware environments (see Table 16). The empirical results demonstrate that even on a low-power edge device (Raspberry Pi 3), the proposed two-stage framework exhibits superior computational efficiency. Compared to the conventional single-stage approach, our architecture reduced processing time from 880 s to 249 s, achieving a 3.5× inference speedup. This significant reduction in latency underscores the "resource-aware" nature of the design, which effectively optimizes limited CPU and memory resources on edge nodes.

Table 16: Inference time comparison across devices for the two-stage model (Test Set: ≈9.33M samples).

Device	PC01	NB01	Edge01
OS	Windows 11	Windows 11	Linux Ubuntu 20.04 LTS
CPU-Type	Intel i5-12500H	Intel i5-1035G4	Cortex-A53
CPU-Speed (GHz)	2.5	1.1	1.5
RAM (GB)	16	8	8
One-Stage Inference Time (s)	183	312	880
Two-Stages Inference Time (s)	51	86	249
Two-Stages Avg. Inference Latency (µs/flow)	5.46	9.21	26.66
Two-Stages Throughput (flows/sec)	183,084	108,573	37,500

To rigorously quantify the real-time capabilities across these platforms, we calculated the Average Inference Latency (T_{lat}) and Throughput (T_{put}) based on the test set size ($N \approx 9.33$M samples) using the equations:

$$(T_{lat}) = \frac{T_{total}}{N_{samples}}, (T_{put}) = \frac{1}{T_{lat}} \tag{8}$$

where T_{total} is the total processing time, and $N_{samples}$ is the number of test samples. Our proposed framework achieves an impressive average latency of 3.53 µs per flow and a throughput of approximately 282,949 flows per second.

As shown in Table 16, the proposed framework maintains ultra-low latency even on resource-constrained hardware. On the Raspberry Pi 3 (Edge01), the average latency is just 26.66 µs per flow, with a throughput of 37,500 flows/sec. This confirms that the model can handle high-velocity IoT traffic in real-time on Cortex-A53 class processors without creating processing bottlenecks.

4.5.2 Real-Time Feasibility in Fog and Distributed Computing

In distributed IoT environments, real-time feasibility is paramount for mitigating attacks before they propagate through the network. Our data indicates that the system can complete packet-level classification for the entire test set in under five minutes. Within a Fog Computing context, this efficiency allows for near-instantaneous intervention—such as triggering automated firewall rules—when high-volume traffic

(e.g., DDoS) is detected. By enabling edge-centric processing, the framework minimizes backhaul bandwidth consumption and round-trip latency, addressing a critical bottleneck in cloud-dependent security models.

4.5.3 Real-Time Inference Analysis and Comparison

To further validate the lightweight nature of the proposed framework, we examined the inference performance gap between a general-purpose PC and a resource-constrained edge device. As shown in Table 16, we compare the average inference latency on PC01 (Intel i5-12500H) and Edge01 (Raspberry Pi 3, Cortex-A53). The results indicate that the proposed model maintains efficient and stable inference behavior across heterogeneous hardware platforms, supporting its suitability for practical edge deployment.

Average latency per flow for the proposed framework on the general-purpose PC (PC01) is 5.46 μs, while in Raspberry Pi (Edge01), it is 26.66 μs. Therefore, while PC01 is about 4.9× as fast, the actual latency of 26.66 μs per flow is low enough to provide real-time processing capabilities for the Raspberry Pi. These latencies correspond to approximately 37,500 flows/second, which is significantly higher than the typical packet arrival rate in standard IoT sensor networks. These data show that the proposed architecture can provide real-time intrusion detection on low-power edge devices with minimal processing load at the edge and without using high-performance computing.

4.5.4 System Scalability and Hierarchical Deployment

The two-stage design provides a distinct architectural advantage for system scalability. Following the principle of hierarchical defense, the framework supports a distributed deployment strategy:

Stage 1 (Lightweight Filtering): Due to the minimal computational footprint of the DT model, it can be hosted natively on localized IoT sensors or lightweight edge nodes for high-speed screening of dominant DDoS traffic.

Stage 2 (Ensemble Analysis): The more complex soft-voting ensemble can be strategically offloaded to fog nodes or edge gateways where relatively higher resources are available. This modularity allows the security infrastructure to scale horizontally as the IoT device density increases, preventing the centralized processing bottlenecks typical of single-stage systems.

4.5.5 Interpretability and Future Extensibility

Beyond performance, the inherent interpretability of the Stage 1 Decision Tree provides network administrators with clear, rule-based logic for security decisions—a prerequisite for "Trustworthy AI" in critical infrastructure. This transparency also facilitates future extensibility toward Federated Learning (FL) and privacy-preserving mechanisms. As identified in recent surveys [10], edge nodes can collaborate to update global model parameters without exposing sensitive raw data, a direction that our lightweight framework is uniquely positioned to support.

5 Conclusions

This paper presents a lightweight, modular two-stage intrusion detection framework for IoT environments. Stage 1 applies mutual information-based feature selection with a fast filter to identify dominant DDoS traffic, while Stage 2 detects minority attacks using a soft-voting ensemble of RF, XGBoost, and AdaBoost. This design reduces computational cost and improves both accuracy and robustness under severe class imbalance. Beyond CICIoT2023, cross-domain validation was conducted on CICIoMT2024 and EdgeIIoTset. The proposed framework achieved over 94.7% accuracy on both datasets, while maintaining competitive detection performance. On a Raspberry Pi 3, the framework processed 10,000 instances per

traffic type in under 2 s (≈1.19 s on average), with per-packet latency between 190.29 µs and 10.84 µs (≈119 µs on average), satisfying the <10 ms real-time constraint for SCADA scenarios. Resource usage remained modest, peaking at 35.78% CPU and 8.45 MB of memory. Overall, the proposed framework delivers stable multi-domain performance, practical edge deployability, and real-time responsiveness on resource-constrained hardware, making it a scalable and adaptable solution for medical, industrial, and general IoT security applications.

Acknowledgement: This work was partly supported by the National Science and Technology Council, Taiwan.

Funding Statement: This work was partly supported by the National Science and Technology Council, Taiwan, under Grant NSTC 114-2221-E-035-065-MY2.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Chung-Wei Kuo; methodology, Chung-Wei Kuo; software, Chung-Wei Kuo and Cheng-Xuan Wu; validation, Chung-Wei Kuo and Cheng-Xuan Wu; formal analysis, Chung-Wei Kuo; investigation, Chung-Wei Kuo and Cheng-Xuan Wu; resources, Chung-Wei Kuo; data curation, Chung-Wei Kuo and Cheng-Xuan Wu; writing—original draft preparation, Chung-Wei Kuo; writing—review and editing, Chung-Wei Kuo and Cheng-Xuan Wu; visualization, Chung-Wei Kuo and Cheng-Xuan Wu; supervision, Chung-Wei Kuo; project administration, Chung-Wei Kuo; funding acquisition, Chung-Wei Kuo. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available in the Canadian Institute for Cybersecurity (CIC) repository at https://www.unb.ca/cic/datasets/iotdataset-2023.html.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sahu SK, Mazumdar K. Exploring security threats and solutions techniques for Internet of Things (IoT): from vulnerabilities to vigilance. Front Artif Intell. 2024;7:1397480. doi:10.3389/frai.2024.1397480.
2. Sinha S. State of IoT 2024: number of connected IoT devices growing 13% to 18.8 billion globally [Internet]. Hamburg, Germany: IoT Analytics; 2024 [cited 2025 Nov 24]. Available from: https://iot-analytics.com/number-connected-iot-devices/.
3. Fatima M, Rehman O, Ali S, Niazi MF. ELIDS: ensemble feature selection for lightweight IDS against DDoS attacks in resource-constrained IoT environment. Future Gener Comput Syst. 2024;159:172–87. doi:10.1016/j.future.2024.05.013.
4. Kamal H, Mashaly M. Enhanced hybrid deep learning models-based anomaly detection method for two-stage binary and multi-class classification of attacks in intrusion detection systems. Algorithms. 2025;18(2):69. doi:10.3390/a18020069.
5. Kumar AG, Rastogi A, Ranga V. Evaluation of different machine learning classifiers on new IoT dataset CICIoT2023. In: Proceedings of the 2024 International Conference on Intelligent Systems for Cybersecurity (ISCS); 2024 May 3–4; Gurugram, India. p. 1–6. doi:10.1109/ISCS61804.2024.10581375.
6. Thereza N, Ramli K. Development of intrusion detection models for IoT networks utilizing CICIoT2023 dataset. In: Proceedings of the 2023 3rd International Conference on Smart Cities, Automation & Intelligent Computing Systems (ICON-SONICS); 2023 Dec 6–8; Bali, Indonesia. p. 66–72. doi:10.1109/ICON-SONICS59898.2023.10435006.
7. Alrefaei A, Ilyas M. Using machine learning multiclass classification technique to detect IoT attacks in real time. Sensors. 2024;24(14):4516. doi:10.3390/s24144516.
8. Ye J, Wang Z, Yang J, Wang C, Zhang C. An LDDoS attack detection method based on behavioral characteristics and stacking mechanism. IoT. 2025;6(1):7. doi:10.3390/iot6010007.

9. Zhang H, Zhang B, Huang L, Zhang Z, Huang H. An efficient two-stage network intrusion detection system in the Internet of Things. Information. 2023;14(2):77. doi:10.3390/info14020077.
10. Hassan SR, Tanveer MU, Prajapat S, Shabaz M. A comprehensive survey on intrusion detection in Internet of Medical Things: datasets, federated learning, blockchain, and future research directions. ICT Express. 2025;11(6):1291–310. doi:10.1016/j.icte.2025.11.005.
11. Gheni HQ, Al-Yaseen WL. Two-step data clustering for improved intrusion detection system using CICIoT2023 dataset. e-Prime Adv Electr Eng Electron Energy. 2024;9:100673. doi:10.1016/j.prime.2024.100673.
12. Hindy H, Atkinson R, Tachtatzis C, Colin JN, Bayne E, Bellekens X. Utilising deep learning techniques for effective zero-day attack detection. Electronics. 2020;9(10):1684. doi:10.3390/electronics9101684.
13. Otoum Y, Nayak A. AS-IDS: anomaly and signature based IDS for the Internet of Things. J Netw Syst Manag. 2021;29(3):23. doi:10.1007/s10922-021-09589-6.
14. Elshweikh AA, Maher AM, Hussein M, Elbayoumy AD. Intrusion detection system for IoT using CICIoT2023 dataset. In: Proceedings of the 2024 International Telecommunications Conference (ITC-Egypt); 2024 Jul 22–25; Cairo, Egypt. p. 326–31. doi:10.1109/NILES63360.2024.10753198.
15. Tseng SM, Wang YQ, Wang YC. Multi-class intrusion detection based on transformer for IoT networks using CIC-IoT-2023 dataset. Future Internet. 2024;16(8):284. doi:10.3390/fi16080284.
16. Neto ECP, Dadkhah S, Ferreira R, Zohourian A, Lu R, Ghorbani AA. CICIoT2023: a real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors. 2023;23(13):5941. doi:10.3390/s23135941.
17. Rai K, Devi MS, Guleria A. Decision tree based algorithm for intrusion detection. Int J Adv Netw Appl. 2016;7(4):2828–34.
18. Suthaharan S. Decision tree learning. In: Machine learning models and algorithms for big data classification: thinking with examples for effective learning. New York, NY, USA: Springer; 2016. p. 237–69. doi:10.1007/978-1-4899-7641-3_10.
19. Zheng D, Hong Z, Wang N, Chen P. An improved LDA-based ELM classification for intrusion detection algorithm in IoT application. Sensors. 2020;20(6):1706. doi:10.3390/s20061706.
20. Albuquerque PHM, do Valle DR, Li D. Bayesian LDA for mixed-membership clustering analysis: the Rlda package. Knowl Based Syst. 2019;163:988–95. doi:10.1016/j.knosys.2018.10.024.
21. Ng AY, Jordan MI. On discriminative vs. generative classifiers: a comparison of logistic regression and naive Bayes. In: Dietterich TG, Becker S, Ghahramani Z, editors. Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS 2001); 2001 Dec 3–8; Vancouver, BC, Canada. p. 841–8.
22. Resende PAA, Drummond AC. A survey of random forest based methods for intrusion detection systems. ACM Comput Surv. 2019;51(3):1–36. doi:10.1145/3178582.
23. Verkerken M, D'hooge L, Wauters T, Volckaert B, De Turck F. Towards model generalization for intrusion detection: unsupervised machine learning techniques. J Netw Syst Manag. 2021;30(1):12. doi:10.1007/s10922-021-09615-7.
24. Bhati BS, Chugh G, Al-Turjman F, Bhati NS. An improved ensemble based intrusion detection technique using XGBoost. Trans Emerg Telecommun Technol. 2021;32(6):e4076. doi:10.1002/ett.4076.
25. Ma M, Zhao G, He B, Li Q, Dong H, Wang S, et al. XGBoost-based method for flash flood risk assessment. J Hydrol. 2021;598:126382. doi:10.1016/j.jhydrol.2021.126382.
26. Shahraki A, Abbasi M, Haugen Ø. Boosting algorithms for network intrusion detection: a comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost. Eng Appl Artif Intell. 2020;94:103770. doi:10.1016/j.engappai.2020.103770.
27. Hossain MA, Saif S, Islam MS. Interpretable machine learning for IoT security: feature selection and explainability in botnet intrusion detection using extra trees classifier. In: Proceedings of the 2024 1st International Conference on Innovative Engineering Sciences and Technological Research (ICIESTR); 2024 May 14–15; Muscat, Oman. p. 1–6. doi:10.1109/ICIESTR60916.2024.10798158.

28. Ali Khan M, Iqbal N, Imran, Jamil H, Kim DH. An optimized ensemble prediction model using AutoML based on soft voting classifier for network intrusion detection. J Netw Comput Appl. 2023;212(3):103560. doi:10.1016/j.jnca.2022.103560.
29. Gueriani A, Kheddar H, Mazari AC, Ghanem MC. A robust cross-domain IDS using BiGRU-LSTM-attention for medical and industrial IoT security. ICT Express. 2025;1–10. doi:10.1016/j.icte.2025.08.011.
30. Akar G, Sahmoud S, Onat M, Cavusoglu Ü, Malondo E. L2D2: a novel LSTM model for multi-class intrusion detection systems in the era of IoMT. IEEE Access. 2025;13:7002–13. doi:10.1109/ACCESS.2025.3526883.
31. Ferrag MA, Friha O, Hamouda D, Maglaras L, Janicke H. Edge-IIoTset: a new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access. 2022;10:40281–306. doi:10.1109/ACCESS.2022.3165809.