STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **16**, October 2026, pp 3210–3233.
Published online in International Academic Press (www.IAPress.org)

An Empirical Comparative Review of Classifiers for Real-Time IoT Cyber Attack Detection

Youssef Boulkhiout [1,*], Samir Balbal [1,2], Khaled Nasri [1], Abdelouahab Moussaoui [1]

[1] *Department of Computer Science, Faculty of Sciences, Setif 1 University – Ferhat ABBAS, Setif, Algeria*
[2] *Artificial Intelligence Laboratory, Faculty of Sciences, Setif 1 University – Ferhat ABBAS, Setif, Algeria*

Abstract The rapid growth of the Internet of Things (IoT) has significantly expanded the attack surface of modern networks, generating an urgent need for robust and efficient intrusion detection mechanisms. Machine learning (ML) is a promising approach to detect cyber attacks in IoT environments by automatically learning discriminative patterns from heterogeneous, real-world traffic. In this work we provide a comparative review and a thorough empirical evaluation of ten classifiers: logistic regression, decision trees, k-nearest neighbors, support vector machines, multilayer perceptrons, Random Forests, XGBoost, LightGBM, CatBoost, and a soft-voting ensemble on the RT-IoT2022 dataset, a realistic benchmark for real-time IoT attack detection. In addition to raw performance, we measure the impact of aggressive dimensionality reduction by comparing the full 83-feature space to a compact subset of 15 features, as determined by the consensus ranking of LightGBM and CatBoost feature importances. Experiments indicate that tree-based ensembles, CatBoost, XGBoost, and LightGBM, achieve the best performance, producing accuracy and AUC values above 99.7% and 99.9%, respectively, for both feature configurations, while maintaining balanced precision and recall across different attack types. Among the non-ensemble models, k-nearest neighbours is the strongest and loses only 0.05 accuracy points under the reduction, and the multilayer perceptron is equally insensitive, whereas logistic regression and the linear SVM lose 16–17 macro-F1 points. All comparisons are supported by stratified five-fold cross-validation, 95% confidence intervals and paired McNemar and Wilcoxon tests under Holm–Bonferroni correction, by a deployment cost profile (training and per-flow inference time), by a per-class analysis of minority-attack recall after SMOTE, and by external validation on NSL-KDD. This paper summarizes the advantages, disadvantages, and deployment considerations of state-of-the-art ML techniques and provides practical guidance for the development of adaptive, lightweight, and interpretable IoT intrusion detection systems.

Keywords Artificial intelligence, Internet of Things security, Machine learning, Multilayer perceptron, Intrusion detection, Anomaly detection

DOI: 10.19139/soic-2310-5070-4092

1. Introduction

The Internet of Things (IoT) has interconnected billions of heterogeneous devices, permeating areas such as smart homes, healthcare, industrial systems, and intelligent transportation. This ubiquitous connectivity brings automation, real-time analytics and improved efficiency, but it also greatly increases the cyber-attack surface. IoT ecosystems are particularly vulnerable to sophisticated threats such as Distributed Denial-of-Service (DDoS) attacks, unauthorized access, malware propagation and zero-day exploits, due to resource-constrained devices, diverse protocols and massive deployment scales.

*Correspondence to: Youssef Boulkhiout (Email: youssef.boulkhiout@ univ-setif.dz). Department of Computer Science, Faculty of Sciences, University of Setif 1 - Ferhat Abbas, Setif, Algeria (19000).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Traditional defenses such as firewalls, signature-based intrusion detection systems (IDS) and static rule sets are ill-suited to the dynamic, polymorphic nature of modern IoT traffic, and do not work at all against unknown attack vectors. As a result, the security community has embraced data-driven approaches that can automatically learn discriminative patterns from complex and high-dimensional traffic. In particular, machine learning (ML) and deep learning (DL) techniques have shown promising results in the detection of known and emerging threats.

A lot of previous work has explored a wide gamut of algorithms, from classical methods (logistic regression, decision trees, support vector machines, k-nearest neighbors, Random Forests, and Gradient Boosting) to state-of-the-art neural architectures, ensemble approaches, federated learning, and semi-supervised paradigms. There are however major challenges such as the choice of models suitable for resource-constrained devices, extreme class imbalance, low-latency inference requirements and robustness to changing attack landscapes.

Motivation and research gap. Four gaps persist in the RT-IoT2022 literature. *(i)* Studies benchmark two to five classifiers drawn from a single model family, so linear, instance-based, kernel, neural and boosted learners have never been compared on this dataset under one preprocessing and evaluation protocol. *(ii)* Reported accuracies exceed 99% and competing methods differ by hundredths of a percentage point, yet cross-validated variability, confidence intervals and paired significance tests are almost never reported, so such differences cannot be separated from sampling noise. *(iii)* Aggressive feature reduction is reported for one chosen model, never as a comparison of how *sensitivity* to that reduction varies across model families – which is precisely what governs the choice of a detector for a constrained gateway. *(iv)* Papers claiming real-time capability seldom report inference cost, so their accuracy rankings cannot be turned into deployment decisions.

Positioning. This work is framed not as a new detection algorithm but as a *practical deployment benchmark*. We evaluate ten heterogeneous classifiers under one protocol, quantify the robustness of each family to an $83 \rightarrow 15$ feature reduction, establish which differences are statistically significant, pair every accuracy figure with its computational cost, and confirm the resulting pattern on a second dataset. It complements recent work in this journal on stacked ensembles [1], feature-optimisation pipelines for reflective denial-of-service detection [2] and the machine-learning design space for intrusion detection [3] by supplying the statistical and cost evidence that turns such comparisons into deployment guidance.

Contributions and novelty. In this paper, we aim to fill these gaps by providing an extensive review and an exhaustive empirical evaluation of state-of-the-art ML/DL techniques on the Real-Time Internet of Things dataset (RT-IoT2022) [4], a newly introduced realistic benchmark for real-time IoT intrusion detection. The main contributions of this work are:

- A structured review of the recent IoT intrusion detection literature, including surveys, classical ML, deep and hybrid architectures, federated and semi-supervised approaches, and comparative benchmark studies;
- A unified empirical comparison of ten classifiers – including linear models, instance-based learners, kernel methods, neural networks, tree-based ensembles and a soft-voting ensemble – within a common preprocessing and evaluation protocol on RT-IoT2022;
- A consensus-based feature selection scheme combining LightGBM and CatBoost importance rankings to reduce the feature space from 83 to 15 attributes, and a systematic analysis of the impact of this reduction on each family of models;
- A statistical validation layer: stratified five-fold cross-validation, 95% confidence intervals, and McNemar and Wilcoxon signed-rank tests under Holm–Bonferroni correction;
- A deployment cost profile – training time and per-flow inference latency – for all ten classifiers under both feature configurations;
- A per-class analysis of minority-attack recall and F1 after SMOTE, and an external validation of the whole pipeline on NSL-KDD;
- Actionable guidance for developing lightweight, adaptive and interpretable IDS that satisfy the stringent performance and resource requirements of modern IoT environments.

The rest of the paper is organized as follows. Related work is reviewed in section 2. The learning algorithms considered are described in section 3. The dataset, the experimental pipeline and the evaluation protocol are described in section 4. section 5 presents and discusses the results, and section 6 outlines the conclusion and future research directions.

2. Related Work

The emergence of IoT devices has brought new security issues and triggered a lot of research on the machine learning based intrusion detection. Table 1 summarizes the studies discussed in this section and reports the used datasets, the methodological approaches and the reported performance. We group this work into six themes, each closed by the limitation it leaves unresolved and mapped onto our design in Section 2.8.

2.1. Surveys and Taxonomies of IoT Threats

Early surveys by Litoussi et al. (2020) [5] and Sobin (2020) [6] identified basic vulnerabilities of IoT ecosystems such as privacy, interoperability and scalability issues and presented taxonomies for IoT architectures, protocols and countermeasures.

Liang and Kim (2021) [7] examined the risks of privacy breaches, DDoS attacks, and architectural vulnerabilities in IoT and advocated the adoption of emerging technologies such as blockchain, edge computing, and ML to enhance security.This perspective was extended by Malhotra et al. (2021) [8] and Shah and Sengupta (2022) [9] who offered wide-ranging surveys of the threats of IoT and IIoT and the various intrusion detection methods in domains such as healthcare and Industry 4.0. Bel and Sabeen (2022) [10] continued to study the security layers of IoT, the main challenges and research directions to reduce cyber risks. Sasi et al. (2024) [11] surveyed IoT attack surfaces and proposed a taxonomy of attack domains, threat categories, and execution methodologies . They discussed countermeasures and gaps in current IoT security and highlighted the need for scalable and adaptive IDS solutions. A recent survey in this journal [3] synthesises the ML and DL design space for intrusion detection.

Limitation. These surveys establish the threat landscape but report no primary experiments, so their call for scalable, adaptive IDS carries no comparable measurements.

2.2. Classical and Deep Learning IDS for IoT

Based on these basic insights, research has evolved towards ML based approaches on IoT IDS. Saba et al. (2022) [12] proposed a CNN based anomaly detection model, which achieved 99.51% and 92.85% accuracy on NID and BoT-IoT datasets respectively, showing the applicability of deep learning in detecting complex attacks. Akash et al. (2023) [13] showed that ensemble classifiers such as Random Forest and XGBoost are able to well capture the non-linear patterns of RT-IoT2022 traffic.Sharmila and Nagapadma (2023) [14] proposed quantized autoencoder for memory and cpu-efficient anomaly detection without compromising on accuracy which can be deployed on edge devices in resource-constrained environments. Rahman et al. [1] showed that stacking heterogeneous learners under one meta-learner outperforms each constituent model.

Limitation. Each study fixes its own preprocessing, imbalance treatment and metrics and commits to a single model family, leaving the families mutually incomparable.

2.3. Class Imbalance and Feature Selection

Hordieieva et al. (2025) [15] compared oversampling techniques, such as SMOTE, GANs, and hybrid methods, for imbalanced datasets. Oversampling found that the GAN-based approaches to generate samples provide more representative samples, thus improving the detection of rare attacks.Moreover, the detection performance has been improved by using feature selection and dimensionality reduction techniques.

Zhang et al. (2024) [16] proposed a two-stage feature selection method using WOA-HA that obtained high accuracy on RT-IoT2022 and Aalto IoT datasets and reduced the feature space by up to 82%. Albalwy and Almohaimeed (2025) [17] showed complementarily that combining feature selection and ANN-based deep learning can further increase the detection performance to 99.7% accuracy. Shaha et al. [2] report a related effect for reflective denial-of-service traffic: a principal-component reduction leaves tree-based classifiers largely unaffected on most sub-datasets while degrading them on one, an early sign that tolerance to reduction depends on both model and data.

Limitation. This literature shows that aggressive reduction is often tolerable for one model, but leaves its family-dependence unquantified and its efficiency gains unmeasured.

2.4. Hybrid and Advanced Deep Architectures

Elzaghmouri et al. (2024) [18] proposed a CNN-BLSTM-GRU-Attention model for detecting advanced attacks such as MQTT Publish flooding, DOS SYN Hping, and NMAP OS detection on RT-IoT2022 with an accuracy of more than 99.6%. Ben Dalla et al. (2025) [19] introduced an adaptive IDS using LSTM for low-computation IoT devices, achieving 98.34% accuracy. Fares et al. (2025) [20] proposed TFKAN, a transformer-based method with Kolmogorov–Arnold Network (KAN) layers that achieves 99.96% accuracy at lower computational costs.Ashraf et al. (2025) [21] further verified the tree-based methods, where Random Forest obtained 99.2% accuracy on the BoT-IoT dataset.

Limitation. These architectures report accuracies within about two points of a well-tuned boosted tree, with no significance test and no cost comparison against that baseline.

2.5. Adversarial Robustness and Zero-day Detection

Bommana et al. (2025) [22] proposed adaptive-weight neural networks with attention mechanisms by combining RBM and RCNN and obtained AUC values ranging from 0.95 to 0.97. Üstebay (2025) [23] combined supervised classification and anomaly detection to identify known and zero-day attacks and achieved 99% accuracy for known attacks and 30-62% for zero-day attacks. Sekhar et al. (2025) [24] used GANs to simulate and defend against threats with 97.62% accuracy on RT-IoT2022, better than conventional IDS approaches.

Limitation. The contrast between 99% on known attacks and 30–62% on zero-day attacks shows that closed-world scores – including ours – are upper bounds, not field estimates.

2.6. Federated and Semi-supervised Frameworks

Attota et al. (2021) [25] proposed a multi-view federated learning framework to improve feature representation and maintain data privacy. Gugueoth et al. (2023) [26] summarized the recent progress of federated and deep learning for IoT security, and discussed advantages of decentralized training. Hamad et al. (2025) [27] surveyed 43 studies on federated learning applications and emphasized the need to develop privacy-preserving IDS. AzharShokoufeh et al. (2025) [28] showed that periodic scheduling in federated frameworks can improve convergence and reduce communication overhead. Alturki and Alsulami (2025) [29] presented semi-supervised learning with entropy-based filtering and demonstrated its robustness on asymmetric IoT networks.

Limitation. These frameworks address orthogonal constraints – privacy, communication cost, label scarcity – and presuppose the choice of local model that this study informs.

2.7. Comparative Benchmarks on RT-IoT2022

Tree-based approaches, such as Random Forest, gradient boosting, and CatBoost are always precise and stable in terms of intrusion detection as stated by Airlangga (2024) [30], Arabiat and Altayeb (2024) [31], Sharmila et al. (2024) [32], and Benmalek and Seddiki (2025) [33]. Gladić et al. (2025) [34] found that using dimensionality reduction techniques improves the efficiency of detection, and enables the design of scalable and high-performance IDS for real-world IoT networks.

Limitation. Closest to ours, yet their accuracies cluster above 99.8%, leaving no headroom for an accuracy-framed contribution, and none reports variability, significance, latency or per-class minority results.

Table 1. Summary of IoT intrusion detection studies, with a gap analysis over six deployment-relevant axes: **#M** (models compared), **FS** (feature configurations), **CV/CI** (variability or confidence intervals), **Sig** (significance test), **Cost** (time/latency/size), **Ext** (multi-dataset); "–" marks axes not applicable to surveys.

Ref	Study	Year	Datasets	Methods/Approach	Accuracy	#M	FS	CV/CI	Sig	Cost	Ext
Surveys and taxonomies of IoT threats (Section 2.1)											
[5]	Litoussi et al.	2020	N/A	Security countermeasures	N/A	–	–	–	–	–	–
[6]	Sobin	2020	N/A	IoT architecture survey	N/A	–	–	–	–	–	–
[7]	Liang and Kim	2021	N/A	Security review	N/A	–	–	–	–	–	–
[8]	Malhotra et al.	2021	N/A	Security challenges survey	N/A	–	–	–	–	–	–
[9]	Shah and Sengupta	2022	N/A	Attack classification survey	N/A	–	–	–	–	–	–
[10]	Bel and Sabeen	2022	N/A	Security survey	N/A	–	–	–	–	–	–
[11]	Sasi et al.	2024	Various	IoT attack survey	N/A	–	–	–	–	–	–
[3]	Azizi et al.	2026	Various	ML/DL for IDS, survey	N/A	–	–	–	–	–	–
Classical and deep learning IDS (Section 2.2)											
[12]	Saba et al.	2022	BoT-IoT, NID	CNN anomaly detection	99.51%, 92.85%	1	1	×	×	×	✓
[13]	Akash et al.	2023	RT-IoT2022	RF, XGBoost, SVM, LR	High	4	1	×	×	×	×
[14]	Sharmila and Nagapadma	2023	RT-IoT2022	Quantized Autoencoder	Maintained	1	1	×	×	✓	×
[1]	Rahman et al.	2025	IDS benchmark	Stacked ensemble (LR, SGD, RF, DNN)	High	5	1	×	×	×	×
Class imbalance and feature selection (Section 2.3)											
[15]	Hordieieva et al.	2025	Network attack data	Oversampling (SMOTE, GANs)	Improved detection	–	1	×	×	×	×
[16]	Zhang et al.	2024	Aalto IoT, RT-IoT2022	WOA-HA feature selection	95.5%, 98.8%	1	2	×	×	×	✓
[17]	Albalwy and Almohaimeed	2025	RT-IoT2022	Feature selection + ANN	99.7%	1	2	×	×	×	×
[2]	Shaha et al.	2025	DRDoS sub-datasets	PCA + tree-based classifiers	High	4	2	×	×	×	✓
Hybrid and advanced deep architectures (Section 2.4)											
[18]	Elzaghmouri et al.	2024	RT-IoT2022	CNN-BLSTM-GRU-Attention	$> 99.6\%$	1	1	×	×	×	×
[19]	Ben Dalla et al.	2025	RT-IoT2022	LSTM-based detection	98.34%	1	1	×	×	✓	×
[20]	Fares et al.	2025	IoT23, CICIoT2023, RT-IoT2022	Transformer + KAN (TFKAN)	99.96%	1	1	×	×	✓	✓
[21]	Ashraf et al.	2025	BoT-IoT	RF, NB, SVM, KNN	99.2% (RF)	4	1	×	×	×	×
Adversarial robustness and zero-day detection (Section 2.5)											
[22]	Bommana et al.	2025	Multiple IoT	Adaptive NN + Attention	AUC: 0.95–0.97	1	1	×	×	×	✓
[23]	Üstebay	2025	CICEVSE2024, CICIoT2023, RT-IoT2022	Supervised + anomaly	99%, 30–62%	2	1	×	×	×	✓
[24]	Sekhar et al.	2025	RT-IoT2022	CGAN + VGG16	97.62%	1	1	×	×	×	×
Federated and semi-supervised frameworks (Section 2.6)											
[25]	Attota et al.	2021	IoT data	Multi-view Federated Learning	Higher	1	1	×	×	×	×
[26]	Gugueoth et al.	2023	N/A	FL/DL review	N/A	–	–	–	–	–	–
[27]	Hamad et al.	2025	N/A	Federated learning review	N/A	–	–	–	–	–	–
[28]	AzharShokoufeh et al.	2025	MNIST, RT-IoT2022	Federated learning	Improved	1	1	×	×	×	✓
[29]	Alturki and Alsulami	2025	CICIoMT2024, CICIoT2023, RT-IoT2022	Semi-supervised learning	High	1	1	×	×	×	✓
Comparative benchmarks on RT-IoT2022 (Section 2.7)											
[30]	Airlangga	2024	RT-IoT2022	GB, RF, LR, MLP	Perfect (RF/GB)	4	1	×	×	×	×
[31]	Arabiat and Altayeb	2024	RT-IoT2022	RF, ANN, LR, SVM	99.9% (RF)	4	1	×	×	×	×
[32]	Sharmila et al.	2024	RT-IoT2022	Decision Trees	99.85%	1	1	×	×	×	×
[33]	Benmalek and Seddiki	2025	RT-IoT2022	PSO + ML/DL	Superior	5	1	×	×	×	×
[34]	Gladić et al.	2025	RT-IoT2022	kNN, RF, XGBoost, MLP, CNN	High	5	1	×	×	×	×
	This work	2026	RT-IoT2022, NSL-KDD	Ten classifiers, consensus feature selection	99.86%	**10**	**2**	✓	✓	✓	✓

2.8. Research Gap and Positioning

Table 1 scores the closest studies on the six axes that decide whether a comparative benchmark can support a deployment decision. Unlike those studies, we compare ten classifier families under one protocol in which the feature configuration is the only manipulated variable, test every difference for significance, pair every accuracy with its cost, and replicate on a second dataset.

3. Methods

In this study, various machine learning algorithms are trained and compared to test their performance on the RT-IoT2022 dataset. The models include gradient boosting, classical machine learning classifiers, a neural network architecture, and ensemble methods. The following subsections give a brief description of each algorithm.

3.1. K-Nearest Neighbours

K-nearest neighbors (KNN) serves as an example-driven, delayed learning algorithm that groups a data point according to the main class amid its 'k' nearest mates in the trait space [35]. It applies distance measures, like Euclidean or Manhattan, to gauge closeness and needs no clear preparation stage. While praised for its ease and natural function, KNN's usefulness depends on proper trait scaling and wise choice of the 'k' factor. Also, its processing cost during guessing can turn prohibitive with large datasets, since it demands matching the test entry to each point in the preparation group [36].

3.2. Support Vector Machine

The support vector machine (SVM) method works by locating the best hyperplane that widens the gap between groups in the trait space. For data not straightly divisible, it uses kernel operations, like polynomial or radial basis function, to map the data into an elevated space where useful division is feasible. Though preparation can demand much processing for big datasets, SVMs are known for their solid broad applicability in elevated contexts. This durability makes them a very useful instrument for diagnostic categorization duties [37].

3.3. Multilayer Perceptron

The multilayer perceptron (MLP) belongs to a type of forward-moving artificial neural network consisting of an entry layer, one or more concealed layers, and an exit layer. Each unit in these layers uses a nonlinear trigger function on a weighted total of its entries, allowing the network to grasp intricate, elevated trait depictions [38]. The model gets prepared through the backpropagation algorithm, which employs gradient-driven refinement to tweak link weights. Owing to their built-in adaptability and ability for nearing complex nonlinear operations, MLPs are very useful for categorization duties, although their effectiveness depends on the presence of ample data and precise factor setup [39].

3.4. Logistic Regression

Logistic regression (LR) is a basic statistical method that has been widely used in two-class and multi-class categorization. It works by estimating the probability of a data entry belonging to a specific group by applying the logistic sigmoid function on a linear combination of input features [40]. The factors of the system are, by maximum likelihood calculation, to find the best match. LR is often the important reference algorithm due to its speedy processing, transparency and good clarity, especially when the basic link between traits and the goal variable is roughly straight [41].

3.5. Decision Tree

Decision trees (DT) are models that partition the dataset several times based on the values of the attributes. This gives rise to a layered structure where the internal nodes represent the decision rules

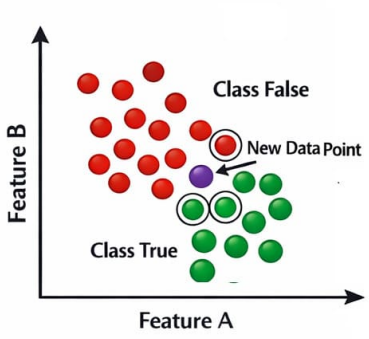

Figure 1. K-Nearest Neighbours

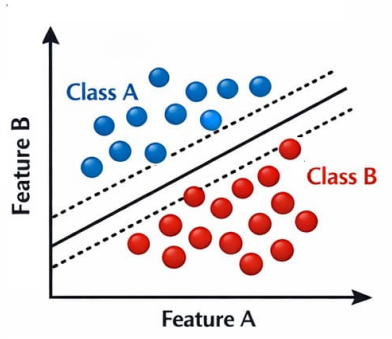

Figure 2. Support Vector Machine

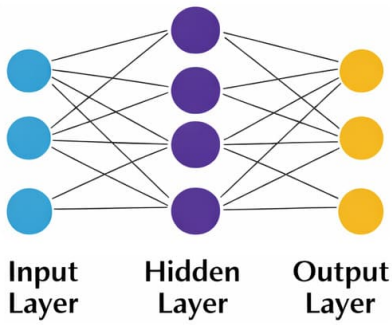

Figure 3. Multilayer Perceptron

and the terminal nodes represent the final outcome. This method can handle diverse data types and understand nonlinear relationships in the data. However, one DT is very likely to overfit to the preparation data. The problem is frequently mitigated and the forecast performance is improved by using decision trees as fundamental building blocks in more powerful groups like Random Forest or various boosting configurations [42].

3.6. Ensemble Learning

Ensemble learning works by combining the predictions of several algorithms to make a better meta-model. For classification this might include a voting mechanism, hard voting based on the majority decision, or soft voting by averaging probabilities [43]. Ensemble learning combines complementary models such as gradient boosting, conventional classifiers, and neural networks to reduce both variance and bias, often outperforming any single model [44].

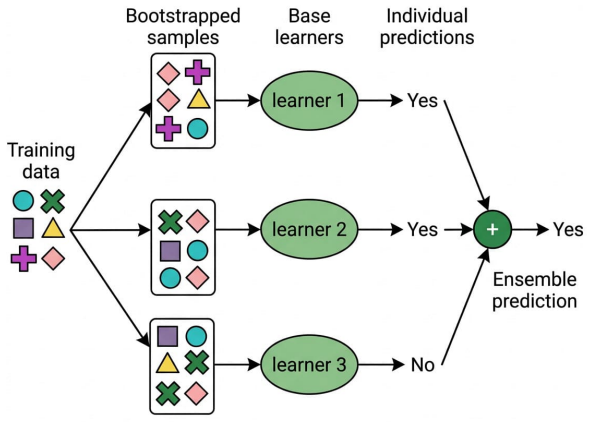

Figure 4. The ensemble learning model

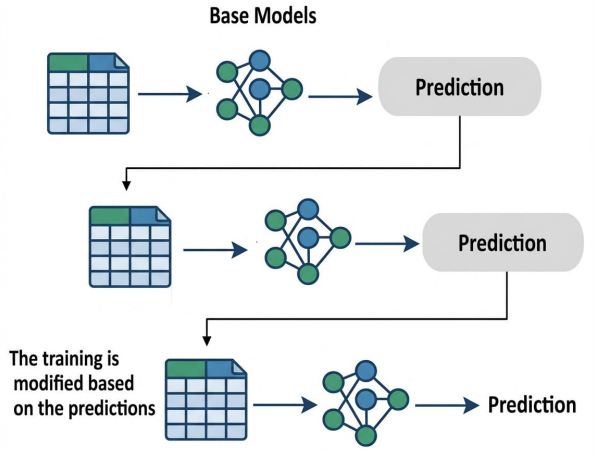

Figure 5. The Boosting Ensemble

3.7. Random Forest

The random forest (RF) method is an ensemble method that works by building many decision trees. Each tree is trained on a different resampled part of the data and uses a random subset of features for each node split. This two-level randomness helps to decouple the distinct trees which reduces the model variation

and successfully limits the overadjustment. All trees contribute their results to the final prediction, which is the majority vote for classification or the average for regression. This powerful framework makes RF particularly suitable for the analysis of complex and noisy health datasets [45].

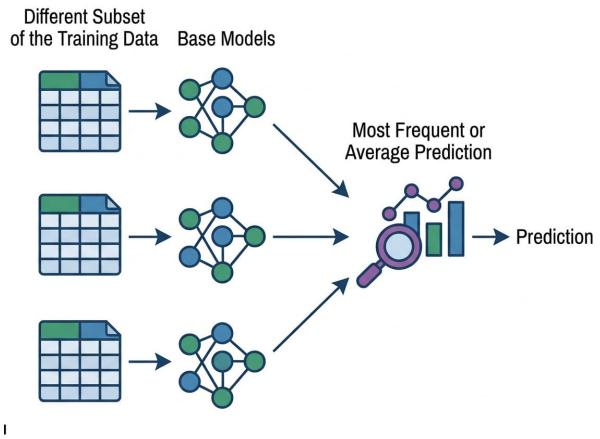

Figure 6. The Bagging Ensemble

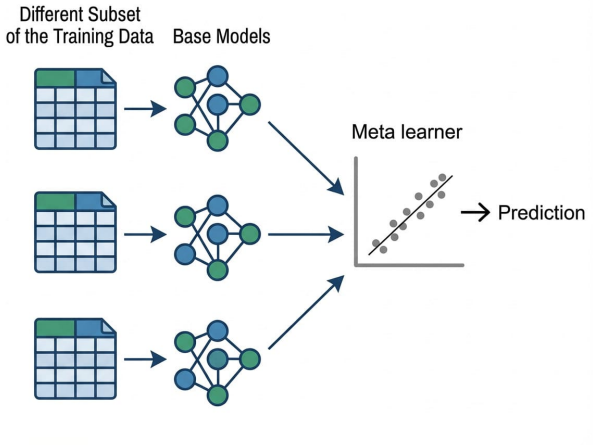

Figure 7. The Stacking Ensemble

3.8. Extreme Gradient Boosting

XGBoost: An improved gradient boosting system that builds the decision trees iteratively to minimize the differentiable loss function. The method combines regularization, shrinkage and feature subsampling to control complexity and improve generalization. It is well suited for large scale structured data problems due to its scalability, parallel computation and distributed processing support. Due to its reliability and performance, it is still one of the most used ML algorithms [46].

3.9. Light Gradient Boosting Machine

LightGBM is an efficient gradient boosting based learning framework which aims to achieve high computational efficiency and predictive accuracy. It uses histogram based decision DT and leaf-wise growth strategy, where a leaf with maximum information gain is grown. This adaptivity enables faster training of the model and less memory consumption than the traditional boosting methods. Also it is very effective for large scale classification problems due to its compatibility to GPU processing and parallel computation [47].

3.10. Categorical Gradient Boosting

CatBoost is a modern boosting framework for efficient processing of categorical data. It builds an ensemble of trees iteratively, where each tree added corrects the residual errors of previous iterations. It reduces overfitting with ordered boosting, and fast inference with symmetric (oblivious) tree structures. Its main advantages - native categorical encoding, shorter training times and better stability - make it a good choice for structured network traffic datasets [48].

4. Experimental Setup

The overall end-to-end pipeline developed for the intrusion detection on the RT-IoT2022 dataset is depicted in Figure 8. The process begins with preprocessing and normalizing all the network traffic

features, so that heterogeneous statistical, temporal, and packet-level attributes are transformed to a comparable scale.Then the dataset is split into two sets of 80% for training and 20% for testing. Stratified sampling is used to keep the proportion of attack and benign instances the same in both subsets.

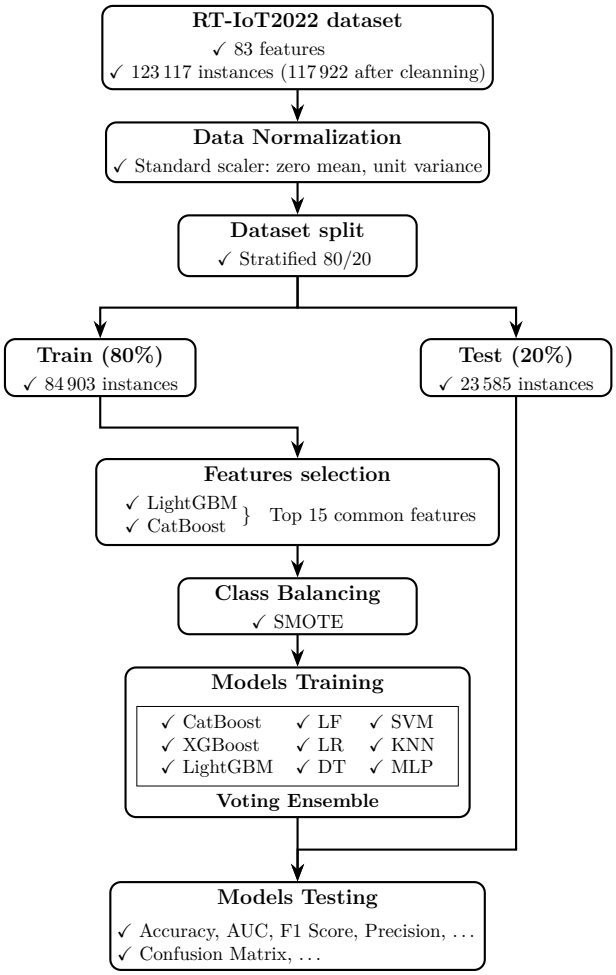

Figure 8. End-to-End Pipeline for Model Evaluation.

Since the class imbalance, typically observed in IoT network traffic, is strong, the training data are balanced by SMOTE-based oversampling or by class weighting during model training. Then, a variety of machine learning algorithms are trained and optimized on the balanced training data: logistic regression, decision trees, k-nearest neighbors, support vector machines, multilayer perceptrons, and several tree-based ensemble methods, namely Random Forest, XGBoost, LightGBM and CatBoost. Moreover, a soft-voting ensemble of these base learners was employed to enhance robustness and generalization ability.

Finally, we assess the performance of all individual models and ensemble approaches on the untouched test dataset. The evaluation is carried out using common intrusion detection metrics such as accuracy, precision, recall, F1-score and AUC-ROC that thoroughly and reliably compare the effectiveness of all evaluated models in detecting different types of IoT attacks.

4.1. Experimental Protocol and Reproducibility

Because the leading classifiers differ by a few hundredths of a point, the order of the pipeline stages determines whether the reported figures are meaningful. Exact duplicate flows are removed first, leaving

117,922 instances; stratified sampling (`random_state` = 42) then yields a 20% test set and an 80% training set, the latter split 90/10 into 84,903 training and 9,434 validation flows, *before every other operation*. The `StandardScaler`, the LightGBM and CatBoost importances and SMOTE are all fitted on the training partition only, so no scaling statistic, importance score or synthetic sample derives from the held-out data, which contains real instances exclusively. A single seed of 42 governs the split, SMOTE, the Random Forest bootstrap and the MLP initialisation.

Hyperparameters were fixed a priori at the values in Table 2 rather than searched: the study compares model *families* under a common budget, following each library's recommended defaults. Class weighting is applied in addition to SMOTE wherever the estimator supports it. Experiments ran under `Python 3.10` with `scikit-learn 1.6.1`, `XGBoost 2.1.4`, `LightGBM 4.6.0`, `CatBoost 1.2.8` and `imbalanced-learn 0.12.4` on a laptop with an Intel Core i7-8665U CPU at 1.90 GHz and 32 GB of RAM.

Table 2. Fixed hyperparameter configuration of each classifier (identical across feature configurations and datasets).

Model	**Configuration**
CatBoost	`iterations`=2000, `depth`=6, `learning_rate`=0.1, `auto_class_weights`=Balanced
XGBoost	`n_estimators`=2000, `max_depth`=6, `tree_method`=hist
LightGBM	`n_estimators`=2000, `max_depth`=−1, `learning_rate`=0.1, `min_child_samples`=10, `class_weight`=balanced
RF	`n_estimators`=2000, `max_depth`=None, `class_weight`=balanced
LR	`solver`=saga, `max_iter`=2000, `class_weight`=balanced
DT	`max_depth`=6, `class_weight`=balanced
SVM	`LinearSVC`(C=1.0, `max_iter`=10000, `class_weight`=balanced) with Platt scaling (`CalibratedClassifierCV`, sigmoid, 5-fold)
KNN	`n_neighbors`=9
MLP	one hidden layer of 100 units, `max_iter`=1000
Voting	soft voting, uniform weights, over the nine base learners above

4.2. Dataset overview

RT-IoT2022 [4] is a dataset collected from a real operation IoT environment, widely used as a benchmark to evaluate security mechanisms in IoT networks. It aggregates network traffic from several cyberattack scenarios and multiple IoT devices, enabling researchers to analyze malicious and benign activities in a real-world context. Since the dataset contains both normal device communication and adversarial behaviors, it captures a large part of the challenges encountered in real IoT deployments. Its main features are shown in Table 3.

Traffic includes devices like ThingSpeak-LED, Wipro-Bulb, MQTT-Temp. Besides the normal operational traffic, a few attacks were intentionally injected that are typical of those that could be targeted at IoT infrastructures, such as brute-force attacks on SSH logins, DDoS attacks using tools such as Hping and Slowloris, and network reconnaissance using Nmap . The availability of benign and attack traffic in the dataset makes it particularly suitable to evaluate and compare intrusion detection approaches in IoT environments.

The class labels used in all figures are the raw strings distributed with RT-IoT2022, including the original spelling `ARP_poisioning`; Table 4 defines each one and reports its frequency before and after oversampling, and all confusion matrices and ROC curves were regenerated with these labels at a legible font size. Three properties govern every later result. The task mixes three benign device classes with nine attack classes – ten in total once the minimum-support filter of Section 4.6 is applied – so accuracy aggregates two decisions of very different difficulty. The imbalance ratio is roughly 3,380 : 1, and always

predicting DOS_SYN_Hping already attains 76.4% accuracy, which is why macro-averaged metrics and per-class recall carry the evidential weight in Section 5. Finally, four of the seven retained attack classes are Nmap variants produced by one tool under different flag settings, hence intrinsically close in flow-feature space.

Table 3. RT-IoT2022 Dataset Characteristics.

Property	**Description**
Dataset Type	Proprietary, real-time IoT infrastructure
Instances	123,117 as distributed; 117,922 after removal of exact duplicate flows
Features	83 as distributed; 81 usable predictors after dropping the row index and the label
Data Structure	Tabular, sequential, multivariate
IoT Devices	ThingSpeak-LED, Wipro-Bulb, MQTT-Temp
Attack Types	Brute-Force SSH, DDoS (Hping, Slowloris), Nmap scans
Traffic Attributes	Bidirectional flows (Zeek + Flowmeter)
Tasks	Classification, regression, clustering
Domain	Engineering / IoT security

Table 4. RT-IoT2022 class labels, source and distribution (SMOTE applied to training only). **FC** (Full Cleaned), **TBSRD** (Train Befor SMOTE & Rare Drop), **TBRD** (Test Before Rare Drop), **TAS** (Train After SMOTE), **T**(Test)

Label	**Type**	**Generating source**	**Full**	FC	TBSRD	TBRD	TAS (used)	T (used)
DOS_SYN_Hping	Attack	TCP SYN flood (hping3)	94,659	90089	64,864	18,018	64,864	18,018
Thing_Speak	Benign	ThingSpeak-LED telemetry	8,108	7,654	5,511	1,531	5,511	1,531
ARP_poisioning	Attack	ARP cache poisoning	7,750	7,625	5,490	1,525	5,490	1,525
MQTT_Publish	Benign	MQTT publish, temperature sensor	4,146	4,142	2,983	828	5,000	828
NMAP_UDP_SCAN	Attack	UDP port scan (nmap -sU)	2,590	2,584	1,860	517	5,000	517
NMAP_XMAS_TREE_SCAN	Attack	Xmas scan (nmap -sX)	2,010	2,010	1,447	402	5,000	402
NMAP_OS_DETECTION	Attack	OS fingerprinting (nmap -O)	2,000	2,000	1,440	400	5,000	400
NMAP_TCP_scan	Attack	TCP connect scan (nmap -sT)	1,002	1,002	722	200	5,000	200
DDOS_Slowloris	Attack	Slow HTTP header exhaustion	534	533	383	107	5,000	107
Wipro_bulb	Benign	Wipro smart-bulb control traffic	253	219	157	44	5,000	44
Excluded: fewer than 100 real training instances								
Metasploit_Brute_Force_SSH	Attack	SSH brute forcing (Metasploit)	37	36	26	7	0	0
NMAP_FIN_SCAN	Attack	Stealth FIN scan (nmap -sF)	28	28	20	6	0	0
	TOTAL		**123,117**	**117,922**	**84,903**	**23,585**	**110,865**	**23,572**

Bidirectional network traffic attributes were logged using Zeek network monitoring tool with Flowmeter plugin, which provide a rich feature space for machine learning applications. The RT-IoT2022 dataset contains 123,117 instances, each described by 83 real-valued and categorical features. The dataset is suitable for a variety of supervised and unsupervised tasks like classification, regression, and clustering across tabular, sequential, and multivariate data structures. RT-IoT2022 is a large scale heterogeneous dataset with complex attack patterns, which makes it an excellent platform to evaluate and improve IDS, and can be leveraged to design robust, adaptive and high performance security solutions for real-time IoT networks.

4.3. Data Normalization

All features of the RT-IoT2022 dataset were initially cast to `float32` precision for numerical stability and consistent feature scaling across heterogeneous network traffic attributes. Then, standardization (z-score normalization) was carried out using scikit-learn's `StandardScaler`. This procedure subtracts the mean

and divides by the standard deviation of each feature, computed only on the training subset to avoid data leakage. Standardization converts each predictor to a distribution with zero mean and unit variance. This prevents features with large numeric ranges from dominating the learning process. This step helps to improve significantly the stability and convergence speed of many ML algorithms. The standardized value z of a raw feature x is given by: $z = \frac{x-\mu}{\sigma}$ where μ and σ denote the mean and standard deviation of the corresponding feature, estimated solely from the training data.

4.4. Data partitioning

The RT-IoT2022 dataset is available as a structured tabular file with 123,117 network flow instances characterized by 83 statistical, temporal and categorical traffic features. We verified that the dataset contained no missing values and then performed a stratified random split into an 80% training set and 20% held-out test set.The distribution between the normal and attack class is highly imbalanced due to the nature of IoT traffic. Stratification keeps this distribution in both subsets. To ensure a reliable and unbiased evaluation of the models, it is important to maintain the same class proportions, especially for the minority attack categories. Figures 9 and 10 show the overall dataset distribution and the distribution of its stratified partitions, respectively. Exact duplicate flows are removed beforehand (123,117 → 117,922), and the training portion is further divided 90/10 into training and validation, giving 84,903 training, 9,434 validation and 23,585 test flows. The resulting class distribution is reported numerically in Table 4.

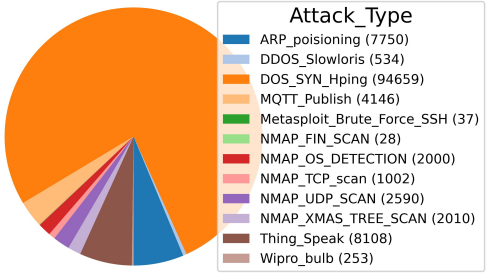

Figure 9. Overall dataset distribution across classes.

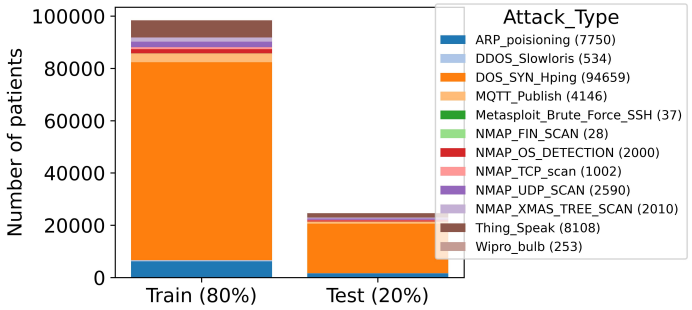

Figure 10. Class distribution across training and test subsets.

4.5. Feature Selection

Feature selection is an important step in intrusion detection systems, since it can improve predictive performance, decrease computational cost and improve the interpretability of the resulting models, which is especially important for real-time IoT environments . In this paper, a hybrid feature selection approach with feature importance was utilized. The method is based on two gradient boosting algorithms, LightGBM and CatBoost, which perform well with high dimensional and heterogeneous tabular data.Both models also produce importance scores for features separately, which indicate the contribution to the predictive performance.

The LightGBM and CatBoost importance vectors, both computed on the training partition only, are min–max normalised onto $[0, 1]$ and combined by the unweighted mean $s_j = \frac{1}{2}(g_j + c_j)$; equal weighting avoids favouring either selector without justification. The two rankings agree only partially, which is what motivates a consensus score rather than either ranking alone. The 15 features with the highest s_j are retained, so the selection requires consistent importance in *both* models and is robust to model-specific bias (Figure 12). The choice of 15 is supported empirically by Figure 13: macro-F1 rises rapidly up to about 15 features and then stays within one standard deviation.

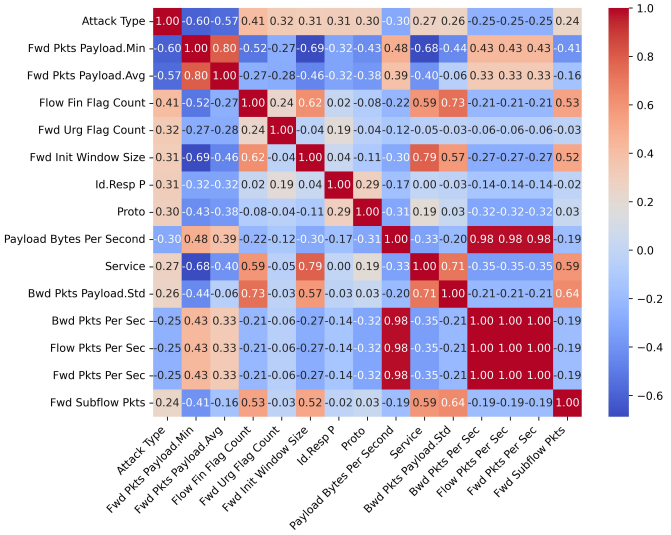

Figure 11. Correlation Heatmap of the 15 Selected Features.

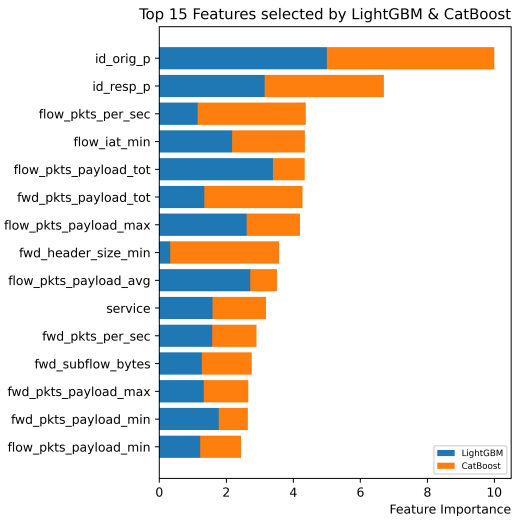

Figure 12. Feature selection results based on feature importance rankings.

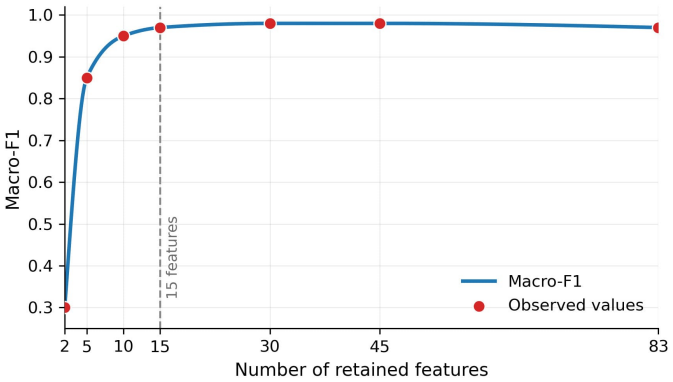

Figure 13. The curve of macro-F1 against the number of retained features (2 to 83).

4.6. Balancing the classes

The RT-IoT2022 dataset exhibits significant class imbalance, where normal traffic instances are much more numerous than some of the attack classes. To solve this problem, Synthetic Minority Over-sampling Technique (SMOTE) was applied only to the training set. SMOTE generates new samples for the minority classes by interpolating between existing samples and their k-nearest neighbors in the feature space, thus reducing the gap between the majority and minority classes while maintaining the original distribution of the validation and test sets. This approach reduces bias of learning algorithms towards dominant classes, improves detection capability of underrepresented attack types, and results in more balanced and reliable evaluation of intrusion detection performance in real-world IoT environments.

A three-threshold class-balancing scheme, with $(\tau_{\min}, \tau_{\text{tgt}}, \tau_{\max}) = (100, 5000, 200000)$, is applied to the training split: classes with fewer than $\tau_{\min}$ samples are removed; classes in $[\tau_{\min}, \tau_{\text{tgt}}]$ are oversampled to τ_{tgt} with SMOTE using $k_{\text{nn}} = \max(1, \min(5, n_{\min} - 1)) = 5$ neighbours; and classes above $\tau_{\max}$ are randomly undersampled. Validation and test sets are restricted to the retained label set but are never resampled (Table 4). For `Wipro_bulb`, SMOTE generates 4,843 synthetic flows from only 157 real training

instances; these lie inside the convex hull of those points and densify a small region rather than diversifying it.

4.7. Performance Evaluation Metrics

The performance of all classifiers was evaluated on the untouched test set using several widely accepted evaluation metrics for multi-class classification including the overall accuracy, macro- and weighted-averaged precision, F1-score, Cohen's Kappa, and the One-vs-Rest (OvR) AUC-ROC. To improve interpretability, several visualization tools were used, such as multiclass confusion matrices, ROC curves, and comparative bar plots of accuracy and AUC-ROC for all models.

- **Confusion Matrix**: A multiclass confusion matrix summarizes the distribution of predicted versus actual classes across all K attack categories:

Table 5. General structure of a multiclass confusion matrix with K classes.

		Predicted Class			
		C_1	C_2	$\cdots$	C_K
True Class	C_1	n_{11}	n_{12}	$\cdots$	n_{1K}
	C_2	n_{21}	n_{22}	$\cdots$	n_{2K}
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	C_K	n_{K1}	n_{K2}	$\cdots$	n_{KK}

Each entry n_{ij} indicates the number of samples belonging to true class C_i that were predicted as class C_j, K the number of classes, and $N = \sum_{i=1}^{K} \sum_{j=1}^{K} n_{ij}$ the total number of samples. One-vs-Rest contingency for each class C_k

$$TP_k = n_{kk}, \quad FP_k = \sum_{\substack{i=1 \\ i \neq k}}^{K} n_{ik}, \quad FN_k = \sum_{\substack{j=1 \\ j \neq k}}^{K} n_{kj}, \quad TN_k = \sum_{\substack{i=1 \\ i \neq k}}^{K} \sum_{\substack{j=1 \\ j \neq k}}^{K} n_{ij} \tag{1}$$

- **Accuracy (ACC)**: Indicates the fraction of instances correctly classified out of the total.

$$ACC = \frac{\sum_{i=1}^{K} n_{ii}}{N} \tag{2}$$

- **F1-Score**: The harmonic average of precision and recall, especially valuable when classes are unevenly distributed. Per-class Precision, Recall and F1-score:

$$\text{Precision}_k = \frac{TP_k}{TP_k + FP_k}, \ \text{Recall}_k = \frac{TP_k}{TP_k + FN_k}, \ F1_k = 2 \times \frac{\text{Precision}_k \cdot \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k} \tag{3}$$

Macro-averaged Precision, Recall, and F1:

$$\text{Precision} = \frac{1}{K} \sum_{k=1}^{K} \text{Precision}_k, \quad \text{Recall} = \frac{1}{K} \sum_{k=1}^{K} \text{Recall}_k, \quad F1 = \frac{1}{K} \sum_{k=1}^{K} F1_k \tag{4}$$

- **AUC-ROC**: Measures the models ability to distinguish between classes across all possible decision thresholds. The ROC curve is constructed by plotting the True Positive Rate (Recall) against the False Positive Rate for each class C_k:

$$TPR_k = \frac{TP_k}{TP_k + FN_k}, \quad FPR_k = \frac{FP_k}{FP_k + TN_k} \tag{5}$$

Let AUC_k be the area under the ROC curve of class C_k in the One-vs-Rest setting. The macro-averaged AUC is:

$$AUC_k = \int_0^1 TPR_k(FPR_k^{-1}(x))\,dx = \int_0^1 TPR_k\,d(FPR_k), \quad AUC = \frac{1}{K}\sum_{k=1}^{K} AUC_k \tag{6}$$

The AUC value represents the area beneath this curve (calculated using the trapezoidal rule), where values closer to 1 reflect stronger discriminative performance.

- **Cohen's Kappa (κ)**: Assesses inter-rater agreement adjusted for chance, especially valuable when classes are unevenly distributed. Values range from -1 to $+1$:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \quad \text{with} \quad p_o = \frac{1}{N}\sum_{i=1}^{K} n_{ii}, \quad p_e = \frac{1}{N^2}\sum_{k=1}^{K}\left(\sum_{j=1}^{K} n_{kj}\right)\left(\sum_{i=1}^{K} n_{ik}\right) \tag{7}$$

4.8. Statistical Validation and Cost Measurement

Classifiers differ by margins as small as 0.013 points, so single-split estimates cannot support a ranking. Every metric is therefore also estimated by stratified five-fold cross-validation on the training partition ($\bar{x} \pm s$, scaler and SMOTE refitted per fold), and every test figure carries a 95% bootstrap percentile interval over 1,000 resamples of the 23,572 held-out flows. Since all models see identical instances, pairs are compared by McNemar's test [49, 50] on the discordant counts n_{01}, n_{10},

$$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \;\sim\; \chi^2_1, \tag{8}$$

with continuity correction (exact binomial test if $n_{01} + n_{10} < 25$). Cross-fold differences in macro-F1 are additionally tested via the Wilcoxon signed-rank test [51], with all p-values Holm–Bonferroni adjusted at family-wise $\alpha = 0.05$. We distinguish statistical from operational significance throughout: at this test-set size, a three-flow difference may reach significance yet remain irrelevant to deployment.

Cost is measured, not assumed. Training time is the wall-clock fitting time on the balanced training partition and testing time is the wall-clock time to score all 23,572 test flows; the mean per-flow latency quoted in Section 5.3 is the latter divided by that count. All measurements are single-threaded, taken after warm-up on an otherwise unloaded machine, and both feature configurations are timed on identical hardware.

4.9. Training Models

All classifiers in this study were trained on the class-imbalance-corrected training set of the RT-IoT2022 dataset. The selected algorithms have already shown to be effective on structured tabular data, as well as the ability to model the complex relations between the 83 network traffic features, such as statistical, temporal and packet level features.After training and hyperparameter optimization of individual models such as logistic regression, decision trees, k-nearest neighbors, support vector machines, multilayer perceptrons, and tree-based ensemble methods (Random Forests, XGBoost, LightGBM, CatBoost), a sophisticated soft-voting strategy was employed to merge the probability outputs of the base learners.

These ensembles improved the predictive accuracy, robustness and generalization across multiple IoT attack types by utilizing the complementary strengths of the individual classifiers, thus provided a reliable framework for real-time intrusion detection in heterogeneous IoT networks.

5. Results and Discussion

This section analyzes the performance of the ten classifiers trained on RT-IoT2022, focusing on misclassification patterns, robustness to class imbalance, and deployment considerations for IoT networks,

using macro-averaged accuracy (ACC), AUC, precision, recall, F1-score, and Cohen's Kappa (κ). The complete numerical results appear in Table 6.

Table 6. Test Performance Metrics (%): 83 vs. 15 Features.

Model	Accuracy		AUC		Precision		Recall		F1-score		Cohen Kappa	
	83	15	83	15	83	15	83	15	83	15	83	15
CatBoost	99.858	99.789	99.995	99.958	97.779	96.133	98.042	97.480	97.728	96.399	99.643	99.470
XGBoost	99.870	99.793	99.999	99.998	97.479	95.347	98.228	97.254	97.673	96.067	99.674	99.480
LightGBM	99.870	99.797	99.976	99.998	98.643	98.099	98.291	97.090	98.369	97.476	99.674	99.490
RF	99.882	99.821	99.276	99.114	96.422	98.230	98.233	97.221	97.212	97.607	99.704	99.551
LR	97.442	94.761	99.494	98.339	76.654	62.224	93.722	81.184	80.687	64.369	93.584	86.870
DT	92.917	96.771	99.050	99.413	73.049	84.492	79.562	89.305	72.806	83.172	82.375	91.926
SVM	98.737	94.294	99.636	98.820	80.330	63.078	96.036	81.636	83.810	66.695	96.832	85.706
KNN	99.740	99.695	98.728	98.520	94.457	94.835	97.333	96.780	95.558	95.430	99.348	99.235
MLP	99.643	99.659	99.510	99.096	91.556	90.662	96.743	96.784	93.638	92.794	99.103	99.144
Voting	99.850	99.801	99.923	99.702	96.296	96.752	98.190	97.264	97.127	96.712	99.623	99.500

The visualization of models performance using ROC curves and confusion matrix are summarized in Figure 15 and Figure 14.

Comparing models trained on the 83-feature set against the reduced 15-feature subset reveals significant differences in robustness, feature dependence and generalization across all metrics. The ensemble and gradient-boosting methods – CatBoost, XGBoost, LightGBM and Voting – are consistently superior across both configurations, with only slight degradation under aggressive feature reduction. With the full feature set these models exceed 99.85% accuracy, and CatBoost, XGBoost and LightGBM exceed 99.97% AUC; with only 15 features they retain more than 99.78% accuracy. The AUC claim above 99.9% excludes Random Forest (99.276%/99.114%) and the 15-feature Voting ensemble (99.702%), whose step-like vote-fraction probabilities limit One-vs-Rest AUC. As Section 5.1 shows, these leading models are not statistically distinguishable and so form one performance tier rather than a ranking. Precision, recall and F1 remain balanced in both configurations and Cohen's Kappa stays above 99.4%, indicating almost perfect agreement beyond chance. This remarkable stability uncovers the core capacity of the boosting-based ensembles to learn non-linear relationships, handle feature interaction, and discard irrelevant or redundant information. Hence, these models are the most reliable and practically feasible solutions for detecting cyber-attacks in real-world IoT.

In both cases, the Multilayer Perceptron (MLP) also performs very well, and its accuracy is marginally higher with the reduced feature set (99.659% vs 99.643%). This margin lies within the fold-to-fold standard deviation of Table 7, so it reflects insensitivity rather than improvement; KNN is in fact the strongest non-ensemble model on both accuracy and macro-F1, the MLP being preferable only where a purely neural, incrementally trainable architecture is required. Traditional machine learning models are, in contrast, much more sensitive to feature reduction.

The performance of Logistic Regression decreases considerably, with a loss of approximately 2.7% in accuracy, and a sharp decrease in precision (from 76.65% to 62.22%) and F1-score (from 80.69% to 64.37%). The SVM is affected most of all ten classifiers, losing 4.44 accuracy points (98.737% $\rightarrow$ 94.294%) and 17.12 macro-F1 points (83.810% $\rightarrow$ 66.695%), whereas KNN is among the most robust: accuracy falls only 0.045 points and macro-F1 0.128, while macro-precision even improves (94.457% $\rightarrow$ 94.835%). Classical models therefore do not degrade uniformly; what degrades is the *global and parametric* subfamily, which fits a single boundary over all coordinates and relies on the redundancy of the 83-dimensional space to compensate for weak individual projections – a compensation lost under reduction. KNN is non-parametric and local, requiring only that the retained coordinates preserve neighbourhood structure, which the consensus subset does by construction; the same locality argument explains the robustness of the tree ensembles and the Decision Tree's improvement. The relevant distinction is the geometry of the decision function, not the age of the algorithm.

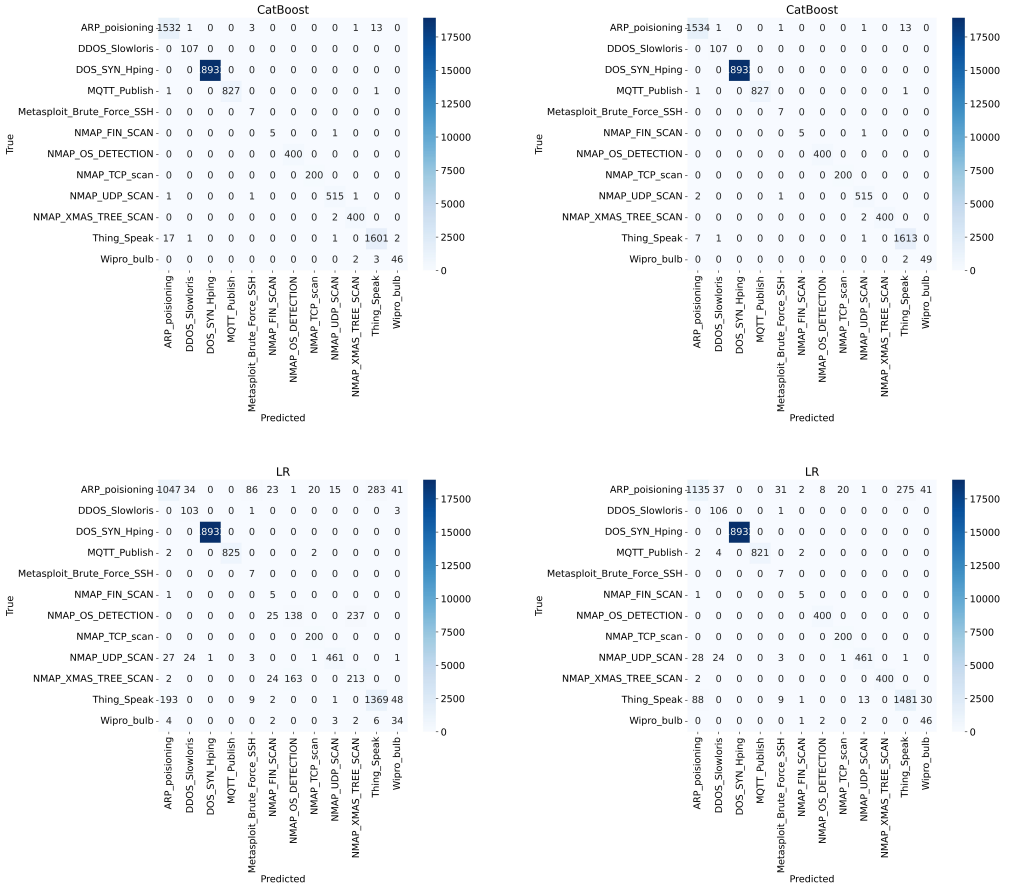

Figure 14. Confusion matrices of the best-performing model (CatBoost, top) and the most feature-sensitive model (LR, bottom) on 15 (left) and 83 (right) features.

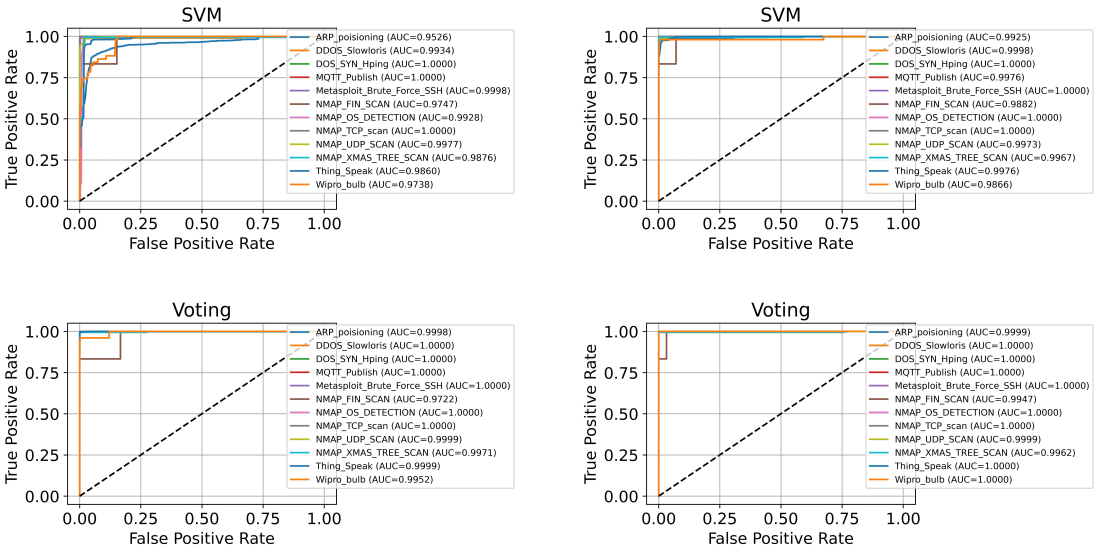

Figure 15. Per-class ROC curves for SVM (top) and the Voting ensemble (bottom) on 15 (left) and 83 (right) features.

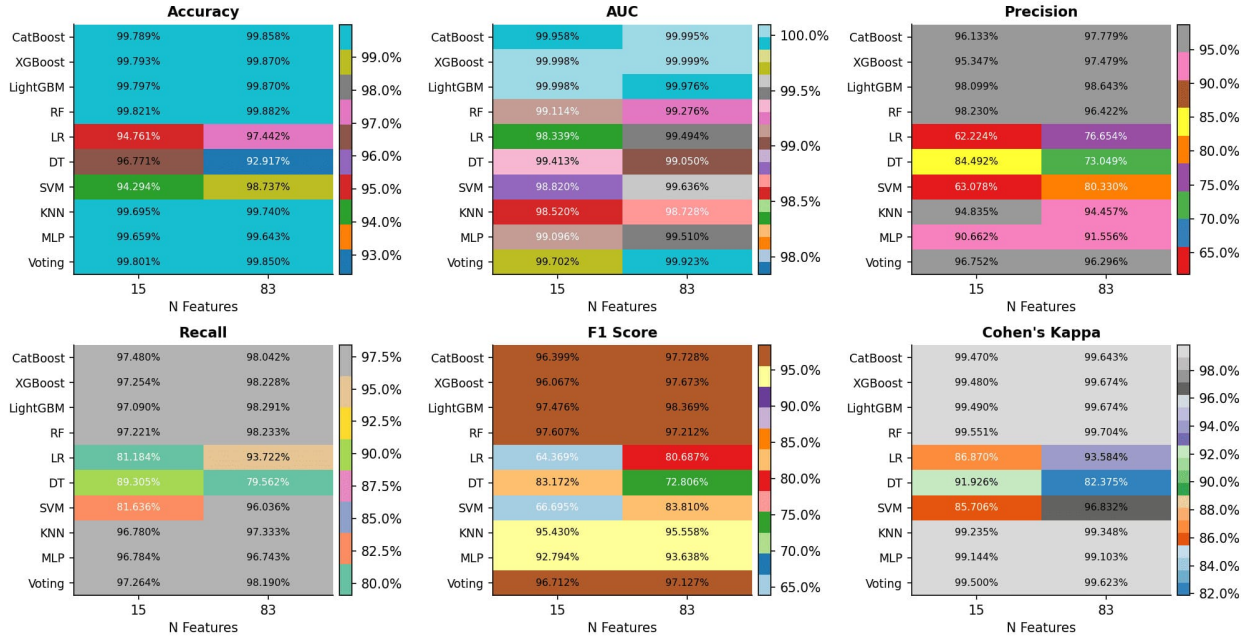

Figure 16. Test heatmaps model × feature Count.

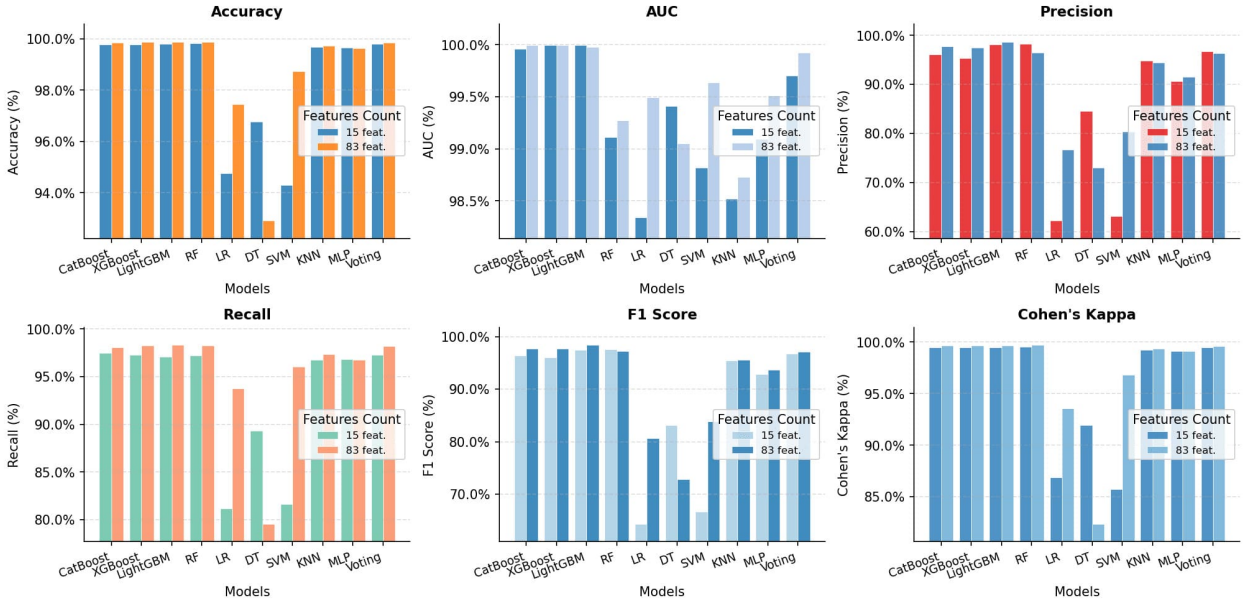

Figure 17. Test metrics by model & feature Count.

The Decision Tree is the one traditional classifier that improves with feature selection: accuracy rises from 92.92% to 96.77%, F1-score from 72.81% to 83.17%, and Cohen's Kappa from 82.38% to 91.93%, indicating that many of the original 83 features add noise for a shallow tree. It nonetheless remains considerably weaker than the best ensemble and neural models.

In summary, the models that perform best with 83 features remain the best with only 15, with no meaningful loss in accuracy or reliability. CatBoost, XGBoost, LightGBM, Random Forest and the Voting ensemble form a single leading tier, with KNN and the MLP close behind at far lower training cost. The extreme dimensionality reduction also brings practical benefits: simpler pipelines, lower computational overhead, faster inference and better suitability for resource-constrained, latency-sensitive IoT deployments.

5.1. Statistical significance

Table 7 reports cross-validated means and bootstrap intervals and Table 8 the paired tests of Section 4.8, both over the label set retained by the minimum-support filter and therefore comparable within Tables 7–9 but not against Table 6; cross-validated macro-averages are omitted since the smallest classes hold only tens of instances per fold.

Table 7. Cross-validation and test-set stability of the 15-feature configuration on RT-IoT2022 (5-fold CV; 95% bootstrap CIs on the 23,572-flow test set).

Model	**CV Acc. (%)**	**CV κ (%)**	**Test Acc. (%) [95% CI]**	**Test macro-F1 (%) [95% CI]**
CatBoost	99.756 ± 0.021	99.399 ± 0.052	99.822 [99.775, 99.873]	99.270 [98.779, 99.636]
XGBoost	99.753 ± 0.030	99.391 ± 0.074	99.792 [99.737, 99.852]	99.309 [98.915, 99.632]
LightGBM	99.763 ± 0.017	99.414 ± 0.043	99.835 [99.784, 99.886]	99.329 [98.795, 99.697]
RF	99.784 ± 0.024	99.467 ± 0.059	99.801 [99.748, 99.856]	99.053 [98.460, 99.510]
LR	95.980 ± 0.718	90.102 ± 1.766	95.690 [95.444, 95.940]	83.319 [81.953, 84.486]
DT	88.438 ± 3.454	71.940 ± 8.278	86.577 [86.134, 86.989]	54.442 [54.124, 54.718]
SVM	94.282 ± 0.091	85.933 ± 0.224	94.158 [93.836, 94.456]	79.482 [78.401, 80.503]
KNN	99.489 ± 0.058	98.740 ± 0.143	99.512 [99.423, 99.587]	96.611 [95.543, 97.489]
MLP	99.290 ± 0.131	98.249 ± 0.323	99.300 [99.188, 99.406]	95.899 [94.849, 96.821]
Voting	99.753 ± 0.022	99.391 ± 0.053	99.809 [99.752, 99.864]	99.235 [98.738, 99.615]

Table 8. Holm–Bonferroni adjusted McNemar tests over all 45 model pairs (RT-IoT2022, 15-feature configuration). n_{01}/n_{10}: discordant counts.

Model A	**Model B**	**Acc. A (%)**	**Acc. B (%)**	n_{01}/n_{10}	$p_{\mathbf{Holm}}$	**Verdict**
Within the leading tier (exact binomial test, $n_{01}+n_{10}<25$)						
LightGBM	CatBoost	99.835	99.822	10 / 7	1.000	not distinguishable
LightGBM	XGBoost	99.835	99.792	16 / 6	0.525	not distinguishable
LightGBM	RF	99.835	99.801	16 / 8	1.000	not distinguishable
LightGBM	Voting	99.835	99.809	11 / 5	1.000	not distinguishable
CatBoost	XGBoost	99.822	99.792	14 / 7	1.000	not distinguishable
CatBoost	RF	99.822	99.801	11 / 6	1.000	not distinguishable
CatBoost	Voting	99.822	99.809	7 / 4	1.000	not distinguishable
XGBoost	RF	99.792	99.801	10 / 12	1.000	not distinguishable
XGBoost	Voting	99.792	99.809	6 / 10	1.000	not distinguishable
RF	Voting	99.801	99.809	6 / 8	1.000	not distinguishable
Leading tier versus {LR, DT, SVM, KNN, MLP}, and within that group						
all 35 remaining pairs					< 0.001	significant

The outcome is unambiguous: none of the ten comparisons within {LightGBM, CatBoost, XGBoost, RF, Voting} survives correction ($p_{\text{Holm}} \geq 0.525$), while all 35 comparisons involving LR, DT, SVM, KNN or MLP are significant at $p_{\text{Holm}} < 0.001$. The choice among the five leading models can therefore be made on cost and interpretability alone, and a reported gain of a few hundredths of a point over them is noise unless a comparable test is supplied – a standard that applies equally to our own 0.013-point margin of LightGBM over CatBoost, a net three flows out of 23,572.

5.2. Minority-class performance after SMOTE

Accuracy is a misleading summary when one class holds 76.4% of the test mass, so Table 9 reports per-class recall and F1 with the absolute test count. Three points follow.

First, the boosters are uniform across classes: LightGBM and CatBoost hold recall at or above 95.4% everywhere, and the gap between CatBoost's macro-F1 (99.27%) and accuracy (99.82%) is carried almost entirely by `Wipro_bulb` and `NMAP_UDP_SCAN`. Second, LR and SVM collapse on exactly those classes: both keep perfect recall on `DOS_SYN_Hping` but fall to 84.09%/77.27% recall on `Wipro_bulb` (F1 40.44%/22.37%) and 51.73%/44.55% on `Thing_Speak`, while on `DDOS_Slowloris` perfect recall pairs with F1 of only 65.24%/64.26% – roughly one false alarm per true detection. Their macro-F1 therefore falls by 16–17 points against only 2.7–4.4 points of accuracy. Third, no claim about the smallest classes is supportable: `Wipro_bulb` has 44 test instances, and `Metasploit_Brute_Force_SSH` and `NMAP_FIN_SCAN` fall below the minimum-support threshold and are excluded.

Table 9. Per-class recall and F1 (%) on the test set (15-feature configuration) for the two leading boosters and two global-parametric classifiers. N_{test}: real test instances.

Class	N_{test}	**LightGBM**		**CatBoost**		**LR**		**SVM**	
		Recall	F1	Recall	F1	Recall	F1	Recall	F1
`DOS_SYN_Hping`	18,018	100.00	100.00	100.00	100.00	100.00	99.99	100.00	99.99
`Thing_Speak`	1,531	98.89	98.79	98.89	98.76	51.73	65.78	44.55	57.24
`ARP_poisioning`	1,525	98.95	98.98	98.95	98.98	85.44	75.25	69.11	62.63
`MQTT_Publish`	828	99.88	99.94	99.88	99.94	99.76	99.88	99.76	99.88
`NMAP_UDP_SCAN`	517	99.61	99.61	99.03	99.22	91.49	89.67	91.88	89.45
`NMAP_XMAS_TREE_SCAN`	402	99.75	99.88	99.75	99.88	99.75	99.88	99.75	99.88
`NMAP_OS_DETECTION`	400	100.00	100.00	100.00	100.00	100.00	99.75	100.00	99.63
`NMAP_TCP_scan`	200	100.00	100.00	99.50	99.75	99.50	97.31	99.00	99.50
`DDOS_Slowloris`	107	100.00	99.53	100.00	99.53	100.00	65.24	100.00	64.26
`Wipro_bulb`	44	95.45	96.55	97.73	96.63	84.09	40.44	77.27	22.37
Macro average	23,572	99.25	99.33	99.37	99.27	91.18	83.32	88.13	79.48

The residual errors concentrate within the Nmap family, where `NMAP_XMAS_TREE_SCAN`, `NMAP_TCP_scan` and `NMAP_UDP_SCAN` come from one tool differing mainly in the flags it sets, so at the flow-aggregate level all present as short, unidirectional, payload-free flows without a completed handshake – exactly the information flow aggregation discards. These confusions are a representational limitation rather than a classifier one. A second, smaller cluster lies between `MQTT_Publish` and `Thing_Speak`; benign/attack confusion matters more operationally, but under LightGBM the three benign classes retain 99.88%, 98.89% and 95.45% recall, so at most 0.12–4.55% of their flows cross that boundary.

5.3. Computational cost

A study claiming real-time relevance must report detector cost. Table 10 gives training and scoring wall-clock times, from which per-flow latency follows by dividing by the 23,572 test flows; these figures resolve the choice among boosters that accuracy cannot. On 15 features LightGBM trains fastest of the four tree ensembles (690 s vs. 745/2,025/2,413 s for XGBoost/RF/CatBoost) but is slowest at inference (887 μs/flow), while CatBoost is the mirror image, most expensive to train yet fastest at inference (21 μs/flow) – which matters more under a per-flow deadline. The Voting ensemble costs the sum of its members (7,006 s, 1,377 μs/flow) while being statistically indistinguishable from the best single model, and is therefore *not* recommended; KNN diverges similarly at 164 μs/flow. The reduction makes these

Table 10. Training and testing time (s), 83- vs. 15-feature configurations.

Model	Train Time (s)		Test Time (s)	
	83	15	83	15
CatBoost	9992.29	2412.61	0.51	0.50
XGBoost	1857.67	745.44	2.37	2.33
LightGBM	4007.94	690.19	29.87	20.92
RF	11032.99	2024.80	7.70	5.07
LR	434.66	68.96	0.16	0.09
DT	24.55	4.47	0.08	0.07
SVM	67694.96	477.17	0.34	0.26
KNN	492.18	20.65	126.44	3.86
MLP	1287.06	393.72	0.42	0.12
Voting	109801.60	7006.18	143.61	32.46

costs tolerable, cutting SVM training from 67,695 s to 477 s and Voting from 109,802 s to 7,006 s, and removing 68 features from the extraction path that usually dominates a gateway pipeline.

5.4. External validation on NSL-KDD

To test whether the observed pattern is specific to RT-IoT2022, the entire pipeline was re-executed without modification on NSL-KDD. The `KDDTrain+` and `KDDTest+` files were pooled and re-partitioned by the same stratified scheme, so this evaluation is closed-world like the first one and its figures are not comparable with results reported on the official open-world test partition; what is compared is the *ordering* of the model families, not the absolute accuracy.

Table 11 reproduces that ordering on the 15-feature configuration. The tree ensembles again lead – LightGBM 98.93%, XGBoost 98.90%, Random Forest 98.83% and Voting 98.72% accuracy, with CatBoost slightly behind at 97.77% – and KNN and the MLP again follow closely (96.87% and 96.66%), confirming that local and axis-aligned learners transfer. The global-parametric classifiers again collapse: the SVM falls to 89.80% accuracy and 68.08% macro-F1, logistic regression to 70.65% and 58.66%.

The single Decision Tree is the one model whose behaviour differs, falling to 45.56% accuracy with a cross-validation standard deviation of 6.17 points, by far the least stable of the ten, which indicates that a depth-6 tree has too little capacity for the 20 retained NSL-KDD classes. The robustness dichotomy is therefore a property of decision geometry rather than an idiosyncrasy of RT-IoT2022.

Table 11. External validation on NSL-KDD under the identical pipeline (15-feature, 28,220 test flows, 20 classes).

Model	Accuracy (%) [95% CI]		AUC (%)	Macro-F1 (%) [95% CI]		Kappa (%)	CV Acc. (%)
CatBoost	97.769	[97.595, 97.932]	99.953	89.038	[88.227, 89.920]	96.450	97.773 ± 0.056
XGBoost	98.899	[98.790, 99.025]	99.974	93.123	[92.300, 93.924]	98.227	98.590 ± 0.155
LightGBM	98.934	[98.821, 99.057]	99.976	93.164	[92.351, 94.017]	98.285	98.558 ± 0.097
RF	98.831	[98.716, 98.963]	99.818	92.560	[91.676, 93.416]	98.116	98.558 ± 0.093
LR	70.647	[70.107, 71.131]	99.095	58.661	[57.810, 59.537]	61.542	70.007 ± 0.628
DT	45.564	[44.991, 46.122]	96.299	51.492	[50.693, 52.180]	38.141	49.451 ± 6.166
SVM	89.796	[89.420, 90.138]	99.029	68.084	[67.019, 69.151]	84.100	89.004 ± 0.326
KNN	96.870	[96.671, 97.074]	98.794	86.248	[85.232, 87.204]	95.027	96.237 ± 0.211
MLP	96.664	[96.465, 96.879]	99.881	85.377	[84.419, 86.434]	94.700	96.088 ± 0.242
Voting	98.717	[98.598, 98.847]	99.965	92.275	[91.472, 93.101]	97.942	98.388 ± 0.085

5.5. Why the families differ, and threats to validity

The pattern is not arbitrary. Gradient-boosted trees lead because the predictive features are threshold-like – a flow is a scan if its duration is short and its packet count small – which an axis-aligned split expresses in one comparison, whereas a linear model must approximate a step function with a hyperplane; boosting further fits the residual at each iteration, concentrating capacity on hard regions such as the Nmap variants, and tree splits are invariant to the heterogeneous feature scales that penalise distance- and gradient-based methods. The three boosters are indistinguishable because they share this bias and differ only in how trees are grown. The SVM degrades most because its decision function aggregates every coordinate into a single margin, so removing coordinates reshapes the geometry rather than merely discarding inputs, an effect compounded by the dense synthetic clusters that inflate the support-vector set; logistic regression degrades for the same reason, a ten-class softmax over 15 features realising only nine linear boundaries. The MLP is unaffected because the reduction cuts its input-layer parameters by 82% while removing only weakly informative directions.

The limitations are stated plainly. The near-perfect scores reflect the benchmark's intrinsic separability as much as detector capability: each attack class was generated by a single tool with fixed parameters, so classes occupy compact, nearly disjoint regions, and the capture is one session on one topology with no distribution shift between partitions. No claim about the two excluded classes is supportable, and SMOTE densifies rather than diversifies classes holding tens of instances. Both evaluations are closed-world with no temporal split, so concept drift is not measured, and flow-level aggregation discards the TCP flags that would separate the Nmap variants. Finally, cost was measured on a laptop rather than gateway hardware and should be read as a relative ranking.

6. Conclusion

This study evaluates ten machine learning models for cyber-attack detection in IoT networks on a comprehensive traffic dataset, using both the full 83-feature set and a reduced 15-feature set. CatBoost, XGBoost, LightGBM, Random Forest and the soft-voting ensemble form a single leading tier – above 99.78% accuracy in both configurations, macro-F1 above 96% and Cohen's Kappa above 99.4% – whose members the paired tests cannot distinguish, so the choice among them is a cost decision rather than an accuracy decision. The three gradient boosters also exceed 99.95% AUC, a claim that extends neither to Random Forest nor to the Voting ensemble. Robustness to the $83 \rightarrow 15$ reduction separates the remaining models by decision geometry rather than by age: KNN and the MLP are essentially unaffected and the single Decision Tree improves markedly, whereas logistic regression and the linear SVM lose 16 and 17 macro-F1 points respectively. These results show that high accuracy and efficiency are achievable with a minimal, discriminative feature set, supporting lightweight, low-latency intrusion detection at the network edge.

Confirmatory versus novel. Three findings replicate known results: boosted trees outperform linear, kernel and single-tree classifiers on tabular data; ensembling reduces variance; and SMOTE improves minority recall. Four are new. Robustness under aggressive reduction depends on decision geometry: *local/axis-aligned* classifiers are largely unaffected while *global/parametric* ones lose 16–17 macro-F1 points. The top models are statistically indistinguishable after correction, so RT-IoT2022 has saturated as an accuracy benchmark. A cost–accuracy frontier then identifies which equivalent model suits a given latency or retraining constraint – the soft-voting ensemble, notably, costs the sum of its members for no measurable gain. NSL-KDD confirms the dichotomy is geometric, not dataset-specific.

Next steps. (i) Use a chronological split on capture timestamps to measure drift sensitivity per family. (ii) Compare a two-stage hierarchical detector (benign vs. attack, then sub-type) against the flat ten-class formulation. (iii) Add per-flow TCP flag distributions to probe the representational error behind Nmap-variant confusion. (iv) Repeat the cost measurements on gateway-class hardware with quantised models. (v) Apply SHAP-based rule extraction and treat rare attacks as a few-shot or anomaly-detection problem.

Acknowledgements

The authors gratefully acknowledge the creators of the RT-IoT2022 dataset for providing open access to this valuable benchmark, which greatly supported the experimental analysis carried out in this work. The authors also thank the open-source machine learning community for developing the tools and frameworks that facilitated the implementation and evaluation of the proposed models. This study was conducted without financial support from any public, commercial, or non-profit funding agencies.

REFERENCES

1. A. Rahman, M. S. Islam Khan, M. Z. A. Eidmum, P. Shaha, B. Muiz, N. Hasan, T. Debnath, D. Kundu, J. T. Tamanna, M. Sayduzzaman, and M. Rahman, *Stacked ensemble method: an advanced machine learning approach for anomaly-based intrusion detection system*, Statistics, Optimization & Information Computing, vol. 14, no. 1, pp. 434–453, 2025.
2. P. Shaha, M. S. Islam Khan, A. Rahman, M. M. Hossain, G. M. Mammun, and M. K. Nasir, *A prevalent model-based on machine learning for identifying DRDoS attacks through features optimization technique*, Statistics, Optimization & Information Computing, vol. 13, no. 1, pp. 409–433, 2025.
3. N. Azizi, A. Jamali, and N. Naja, *Comprehensive study: machine learning and deep learning approaches in intrusion detection systems*, Statistics, Optimization & Information Computing, vol. 15, no. 2, pp. 1416–1432, 2026.
4. B. S. and R. Nagapadma, *RT-IoT2022 (Real Time Internet Of Things)*, UCI Machine Learning Repository, 2023.
5. M. Litoussi, N. Kannouf, K. El Makkaoui, A. Ezzati, and M. Fartitchou, *IoT security: challenges and countermeasures*, in Proceedings of the 7th International Symposium on Emerging Information, Communication and Networks (EICN 2020), pp. 503–508, 2020.
6. C. C. Sobin, *A Survey on Architecture, Protocols and Challenges in IoT*, Wireless Personal Communications, 2020.
7. X. Liang, and Y. Kim, *A Survey on Security Attacks and Solutions in the IoT Network*, in 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), pp. 853–859, 2021.
8. P. Malhotra, Y. Singh, P. Anand, D. K. Bangotra, P. K. Singh, and W.-C. Hong, *Internet of Things: Evolution, Concerns and Security Challenges*, Sensors, vol. 21, no. 1809, 2021.
9. Y. Shah, and S. Sengupta, *A Survey on Classification of Cyber-attacks on IoT and IIoT devices*, Webology, vol. 19, no. 1, pp. 3741–3756, 2022.
10. H. J. F. Bel, and S. Sabeen, *A Survey on IoT Security: Attacks, Challenges and Countermeasures*, Webology, vol. 19, no. 1, pp. 3741, 2022.
11. T. Sasi, A. H. Lashkari, R. Lu, P. Xiong, and S. Iqbal, *A Comprehensive Survey on IoT Attacks: Taxonomy, Detection Mechanisms and Challenges*, Journal of Information and Intelligence, vol. 2, pp. 455–513, 2024.
12. T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, *Anomaly-based intrusion detection system for IoT networks through deep learning model*, Computers and Electrical Engineering, vol. 99, pp. 107810, 2022.
13. T. R. Akash, Md. S. A. Sourav, S. A. Sarna, and Md. Rakibuzzaman, *Develop Automated Systems that Gather and Analyze Threat Data to Protect Business Systems Automatically from Cyberattacks*, American Journal of Engineering, Mechanics and Architecture, vol. 1, no. 6, pp. 90–100, 2023.
14. B. S. Sharmila, and R. Nagapadma, *Quantized autoencoder (QAE) intrusion detection system for anomaly detection in resource-constrained IoT devices using RT-IoT2022 dataset*, Cybersecurity, vol. 6, pp. 41, 2023.
15. V. Hordieieva, I. Czarnowski, and P. Jedrzejowicz, *A Comparative Study of Oversampling Techniques for Imbalanced Network Attack Detection*, Procedia Computer Science, vol. 270, pp. 822–830, 2025.
16. K. Zhang, Y. Liu, X. Wang, F. Mei, G. Sun, and J. Zhang, *Enhancing IoT feature selection: A two-stage approach via an improved whale optimization algorithm*, Expert Systems With Applications, vol. 256, pp. 124936, 2024.
17. F. Albalwy, and M. Almohaimeed, *Advancing Artificial Intelligence of Things Security: Integrating Feature Selection and Deep Learning for Real-Time Intrusion Detection*, Systems, vol. 13, pp. 231, 2025.
18. B. M. Elzaghmouri, Y. H. F. Jbara, S. Elaiwat, N. Innab, A. A. F. Osman, M. A. M. Ataelfadiel, F. H. Zawaideh, M. F. Alawneh, A. Al-Khateeb, and M. Abu-Zanona, *A Novel Hybrid Architecture for Superior IoT Threat Detection through Real IoT Environments*, Computers, Materials & Continua, 2024.
19. L. O. Ben Dalla, O. Karal, and A. Degirmenciyi, *Leveraging LSTM for Adaptive Intrusion Detection in IoT Networks: A Case Study on the RT-IoT2022 Dataset implemented On CPU Computer Device Machine*, in 5th International Conference on Engineering, Natural and Social Sciences, Konya, Turkey, pp. 1–10, 2025.
20. I. A. Fares, M. Abd Elaziz, A. O. Aseeri, H. S. Zied, and A. G. Abdellatif, *TFKAN: Transformer based on Kolmogorov–Arnold Networks for Intrusion Detection in IoT environment*, Egyptian Informatics Journal, vol. 30, pp. 100666, 2025.
21. J. Ashraf, G. M. Raza, B.-S. Kim, A. Wahid, and H.-Y. Kim, *Making a Real-Time IoT Network Intrusion-Detection System (INIDS) Using a Realistic BoT–IoT Dataset with Multiple Machine-Learning Classifiers*, Applied Sciences, vol. 15, no. 2043, 2025.
22. S. R. Bommana, S. Veeramachaneni, S. E. Ahmed, and M. B. Srinivas, *Addressing Adversarial Attacks in IoT Using Deep Learning AI Models*, IEEE Access, vol. 13, pp. 50437–50452, 2025.
23. S. Üstebay, *Enhancing Zero-Day Attack Detection in IoT Networks via Isolation Forest and Ensemble Tree Models*, Electrica, vol. 25, pp. 1–8, 2025.

24. M. C. Sekhar, V. Alukapelly, T. Neelima, R. Kotoju, and V. Veerabhadram, *Generative Adversarial Networks for Cyber Threat Simulation and Defence Strategies*, Journal of Theoretical and Applied Information Technology, vol. 103, no. 4, pp. 1–15, 2025.
25. D. C. Attota, V. Mothukuri, R. M. Parizi, and S. Pouriyeh, *An Ensemble Multi-View Federated Learning Intrusion Detection for IoT*, IEEE Access, vol. 9, pp. 117734–117746, 2021.
26. V. Gugueoth, S. Safavat, and S. Shetty, *Security of Internet of Things (IoT) using federated learning and deep learning — Recent advancements, issues and prospects*, ICT Express, vol. 9, pp. 941–960, 2023.
27. N. A. Hamad, K. A. Abu Bakar, F. Qamar, A. M. Jubair, R. R. Mohamed, and M. A. Mohamed, *Systematic Analysis of Federated Learning Approaches for Intrusion Detection in the Internet of Things Environment*, IEEE Access, vol. 13, pp. 95410–95429, 2025.
28. D. AzharShokoufeh, N. DerakhshanFard, F. RashidJafari, and A. Ghaffari, *Optimizing IoT data collection through federated learning and periodic scheduling*, Knowledge-Based Systems, vol. 317, pp. 113526, 2025.
29. B. Alturki, and A. A. Alsulami, *Semi-Supervised Learning with Entropy Filtering for Intrusion Detection in Asymmetrical IoT Systems*, Symmetry, vol. 17, pp. 973, 2025.
30. G. Airlangga, *Comparative Analysis of Machine Learning Models for Intrusion Detection in Internet of Things Networks Using the RT-IoT2022 Dataset*, MALCOM: Indonesian Journal of Machine Learning and Computer Science, vol. 4, no. 2, pp. 656–662, 2024.
31. A. Arabiat, and M. Altayeb, *Enhancing internet of things security: evaluating machine learning classifiers for attack prediction*, International Journal of Electrical and Computer Engineering (IJECE), vol. 14, no. 5, pp. 6036–6046, 2024.
32. B. S. Sharmila, B. M. Nandini, S. S. Kavitha, and A. Srivatsa, *Performance Evaluation of Parametric and Non-Parametric Machine Learning Models using Statistical Analysis for RT-IoT2022 Dataset*, Journal of Scientific & Industrial Research, vol. 83, pp. 864–872, 2024.
33. M. Benmalek, and A. Seddiki, *Particle swarm optimization-enhanced machine learning and deep learning techniques for Internet of Things intrusion detection*, Journal of Cybersecurity, 2025.
34. D. Gladić, J. Petrovački, S. Sladojević, M. Arsenović, and S. Ristić, *Analysis of different IDS-based machine learning models for secure data transmission in IoT networks*, Open Computer Science, vol. 15, pp. 20250032, 2025.
35. T. Cover, and P. Hart, *Nearest neighbor pattern classification*, IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21–27, 1967.
36. N. S. Altman, *An introduction to kernel and nearest-neighbor nonparametric regression*, The American Statistician, vol. 46, no. 3, pp. 175–185, 1992.
37. C. Cortes, and V. Vapnik, *Support-vector networks*, Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
38. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning representations by back-propagating errors*, Nature, vol. 323, no. 6088, pp. 533–536, 1986.
39. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 2002.
40. D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*, John Wiley & Sons, 2013.
41. D. R. Cox, *The regression analysis of binary sequences*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 20, no. 2, pp. 215–232, 1958.
42. J. R. Quinlan, *Induction of decision trees*, Machine Learning, vol. 1, no. 1, pp. 81–106, 1986.
43. T. G. Dietterich, *Ensemble methods in machine learning*, in International Workshop on Multiple Classifier Systems, Springer, pp. 1–15, 2000.
44. R. Polikar, *Ensemble based systems in decision making*, IEEE Circuits and Systems Magazine, vol. 6, no. 3, pp. 21–45, 2006.
45. L. Breiman, *Random forests*, Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
46. T. Chen, and C. Guestrin, *XGBoost: A scalable tree boosting system*, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, 2016.
47. G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, *LightGBM: A highly efficient gradient boosting decision tree*, in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.
48. L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, *CatBoost: unbiased boosting with categorical features*, in Advances in Neural Information Processing Systems (NeurIPS), vol. 31, 2018.
49. Q. McNemar, *Note on the sampling error of the difference between correlated proportions or percentages*, Psychometrika, vol. 12, no. 2, pp. 153–157, 1947.
50. T. G. Dietterich, *Approximate statistical tests for comparing supervised classification learning algorithms*, Neural Computation, vol. 10, no. 7, pp. 1895–1923, 1998.
51. J. Demšar, *Statistical comparisons of classifiers over multiple data sets*, Journal of Machine Learning Research, vol. 7, pp. 1–30, 2006.