An efficient ensemble tree-based framework for intrusion detection in industrial internet of things networks

Mouad Choukhairi[1], Oumaima Chentoufi[2], Ouail Choukhairi[1], Youssef Fakhri[1]
[1]LARI Laboratory, Department of Computer Science, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco
[2]Engineering Science Laboratory, National School of Applied Sciences (ENSA), Ibn Tofail University, Kenitra, Morocco

Article Info

Article history:

Received Apr 27, 2025
Revised Oct 31, 2025
Accepted Nov 8, 2025

Keywords:

Cybersecurity
Ensemble learning
IIoT security
Intrusion detection
Machine learning
Multiclass
ToN-IoT

ABSTRACT

The increasing complexity of cyber threats in industrial internet of things (IIoT) environments necessitates robust, scalable, and efficient intrusion detection systems (IDS). This study presents a novel ensemble tree-based framework that integrates gradient boosting-based machine learning models, including XGBoost, LightGBM, AdaBoost, and CatBoost, with mutual information (MI) feature selection and synthetic minority over-sampling technique (SMOTE) to enhance multiclass intrusion detection performance. The framework is designed to handle large-scale, imbalanced datasets efficiently while maintaining high classification accuracy. Performance evaluation using the telemetry of network (ToN)-IoT benchmark dataset demonstrates that the proposed models achieve a high accuracy of 99.43%, with a strong precision-recall balance and an F1-score, ensuring minimal false positive rates of 0.08%. By leveraging MI for optimal feature selection and SMOTE for data balancing, this approach effectively enhances detection capabilities in highly dynamic network environments. The lightweight architecture and reduced execution time make the framework well-suited for deployment in edge or fog nodes within smart industrial environments. The proposed solution provides a scalable and adaptable methodology for securing IIoT networks, making it applicable for real-time intrusion monitoring and further cybersecurity advancements in industrial systems.

Corresponding Author:

Mouad Choukhairi
LARI Laboratory, Department of Computer Science, Faculty of Sciences, Ibn Tofail University
B.P 133, University Campus, Kenitra, Morocco
Email: mouad.choukhairi@uit.ac.ma

1. INTRODUCTION

The industrial internet of things (IIoT) is transforming modern industries by seamlessly integrating sensors, actuators, and control systems, thereby facilitating real-time data exchange and enabling unprecedented levels of operational automation [1]. This interconnected ecosystem allows for enhanced monitoring, predictive maintenance, and optimized resource allocation, leading to increased efficiency and productivity across various sectors [2]. However, this increased connectivity inherently introduces new vulnerabilities, making critical infrastructures more susceptible to sophisticated cyberattacks [3]. Traditional intrusion detection systems (IDS) often fall short in effectively safeguarding IIoT networks due to the dynamic nature of IIoT data and the continuous emergence of novel, zero-day exploits [4], [5]. Signature-based IDS, which rely on predefined attack patterns, struggle to detect anomalies and deviations from established norms in these complex environments. The inadequacy stems from their inability to adapt to the evolving threat landscape and the unique characteristics of IIoT traffic patterns.

IIoT datasets present unique challenges for machine learning (ML) based IDS, including high dimensionality, class imbalance, and inherent noise, which significantly complicates the training and deployment of effective detection models [6]. The high dimensionality of IIoT data, characterized by a large number of features extracted from network traffic and sensor readings, can lead to the curse of dimensionality, where the performance of ML algorithms degrades as the number of features increases. Class imbalance, where the number of instances belonging to different attack classes varies significantly, further exacerbates the problem, as ML models tend to be biased towards the majority class, resulting in poor detection rates for minority classes, which often represent critical security threats.

Furthermore, the presence of noise in IIoT data, arising from sensor inaccuracies, communication errors, and environmental factors, can further degrade the performance of ML models, leading to increased false positive rates (FPR) and reduced detection accuracy. To address the challenge of detecting anomalies and unknown attacks in real-time within IoT devices, ML techniques can be leveraged [7]. The application of ML algorithms offers the potential for automating anomaly detection in industrial machinery by analyzing the vast amounts of data generated by IoT devices [8].

ML models have demonstrated significant promise in the realm of IDS for IIoT, yet they also present certain limitations that need to be carefully addressed. The effectiveness of IDS has grown in popularity recently, and identifying unauthorized individuals is its main objective [9]. Ensemble ML models have shown remarkable performance in intrusion detection tasks due to their ability to combine multiple base learners and capture complex relationships within the data [10]. However, even with tree-based algorithms, attackers can introduce small changes in IoT network traffic that can mislead these algorithms. Despite the increasing research efforts, anomaly detection using ML is still evolving [7]. The current ML models lack robustness when facing previously unseen types of attacks [11]. The models' ability to generalize across diverse IIoT environments and adapt to evolving attack strategies remains a concern. Thus, new attack detection methods are needed for risk mitigation [12].

Advanced methods are needed because traditional approaches for detecting cyber-attacks have low efficiency [13]. Therefore, there is still the opportunity to develop effective intrusion detection for large-scale IIoT systems. Gradient boosting ML algorithms like XGBoost, LightGBM, AdaBoost, and CatBoost have gained attention due to their ability to capture non-linear relationships and scale to large datasets with high-performance learning. However, their performance in IIoT scenarios is constrained by challenges such as high feature dimensionality and class imbalance, which can lead to biased models or increased false alarms.

To address these limitations and challenges, this paper proposes a comprehensive framework that integrates an ensemble tree-based architecture consisting of XGBoost, LightGBM, AdaBoost, and CatBoost as state-of-the-art gradient boosting classifiers with mutual information (MI) for feature selection and synthetic minority over-sampling technique (SMOTE) for class balancing. The novelty of this research lies in combining MI and SMOTE with four popular gradient boosting classifiers in a unified IDS pipeline. Unlike previous studies that evaluate only individual components or models, we systematically benchmark multiple models' scenarios, analyze the interaction of pre-processing strategies, and provide execution time analysis to determine real-time feasibility. Evaluation on the telemetry of network (ToN)-IoT dataset demonstrates that our approach attains high classification accuracy while preserving low FPR and efficient runtimes, making it viable for real-time IIoT intrusion monitoring.

2. RELATED WORK

In the realm of IIoT intrusion detection, feature engineering and class balancing strategies are pivotal in addressing the challenges posed by high-dimensional and imbalanced datasets. Feature engineering strategies have been extensively explored to enhance detection accuracy, reduce false positives, and manage high-dimensional data. MI is a prominent technique used for feature selection, which helps reduce redundancy and select the most relevant features, thereby improving classification accuracy and detection performance in IIoT networks [14], [15]. Principal component analysis (PCA) is another widely used method for feature extraction, which has been shown to significantly improve detection accuracy, achieving up to 100% in some cases by transforming high-dimensional data into a lower-dimensional space while retaining essential information [16], [17]. Relief is a known feature selection method that evaluates the importance of features based on their ability to distinguish between different classes [18]. The light feature engineering based on the mean decrease in accuracy (LEMDA) method, a novel feature engineering approach, has demonstrated a substantial improvement in F1-scores by an average of 34% across various models, indicating its effectiveness in enhancing model performance while reducing training and detection times [19]. Additionally, bio-inspired feature selection methods like gray wolf optimization (GWO) have been shown to outperform other techniques, achieving high accuracy and F1-scores with reduced execution time when combined with classifiers like k-nearest neighbors (KNN) [20]. The integration of feature selection and reduction techniques, such as minimum redundancy maximum relevance and PCA, has been effective in balancing model complexity and performance, achieving high accuracy rates of up to 99.9% in binary classification tasks [21].

In addressing class imbalance in the same context, various studies have explored the effectiveness of different class-balancing strategies, such as SMOTE, adaptive synthetic sampling (ADASYN), and other oversampling and undersampling techniques. SMOTE is frequently highlighted for its ability to enhance classification performance by producing synthetic samples for the minority class, thereby improving metrics like F1-score, precision, and recall. For instance, in one study, SMOTE achieved a precision of 99.19%, a recall of 72.45%, and an F1-score of 79.13% when applied to the IoT-23 dataset, indicating a balanced improvement in detection performance [22]. ADASYN, which adapts the number of synthetic samples generated for different minority class examples based on their difficulty, has also been shown to improve classification metrics, although specific performance metrics were not detailed in the provided contexts [23]. Other studies have compared these techniques with ensemble models, finding that methods like SMOTE, when combined with ensemble learners, can significantly boost accuracy by 1% to 4% and achieve precision, recall, and F1-scores between 95% and 100% [24]. Additionally, the integration of these techniques with advanced models like XGBoost has demonstrated remarkable effectiveness, achieving F1-scores as high as 99.9% on imbalanced IIoT datasets [25]. Despite these improvements, challenges remain, as oversampling and undersampling can sometimes lead to high false-positive rates or reduced performance in majority classes, necessitating further refinement and hybrid approaches [25], [26].

3. METHOD

This section details the workflow adopted to build, design, and assess the proposed intrusion detection framework. The pipeline is arranged in five sequential blocks: data preparation and pre-processing, feature engineering, class balancing, model development, and evaluation. All steps were executed in the sequence presented and were designed to enable full reproducibility. Figure 1 gives a high–level overview, while Algorithm 1 lists the exact steps mirrored in our approach.

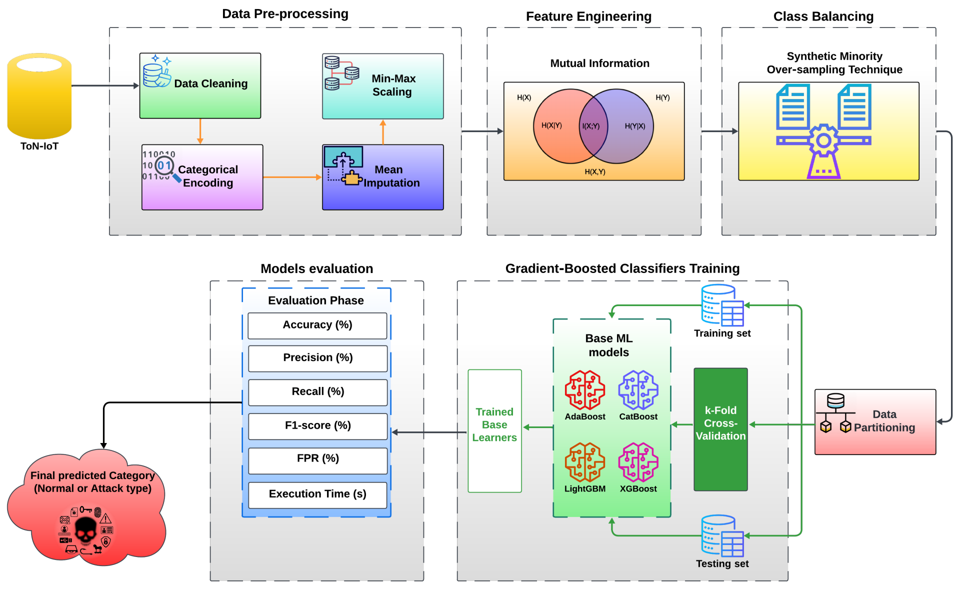

Figure 1. Framework of the proposed workflow for cyberattack classification in IIoT networks

3.1. Data preparation and pre-processing

The ToN-IoT dataset is a next-generation benchmark expressly crafted for IoT and IIoT cybersecurity research [27]. Built in an Industry 4.0 cyber-range at UNSW Canberra, it fuses time-aligned telemetry from more than ten industrial sensors, yielding millions of records that are individually labelled as benign or as one of nine representative attack families (i.e., denial of service, ransomware, man in the middle, password/brute-force, distributed denial of service, backdoor, injection, cross-site scripting, and scanning). This multimodal design mirrors the cloud–fog–edge hierarchy typical of modern factories, letting researchers test AI-driven intrusion-detection and threat-intelligence models under realistic IIoT traffic and class-imbalance conditions. Consequently, ToN-IoT has become a de facto reference corpus for evaluating security analytics in Industry 4.0 environments.

To enable the classification of IoT-based cyberattacks, a total of forty-three features are extracted to characterize each flow, categorized into six subsets based on the nature of the information they convey (e.g., connection activity features, violation activity features, and statistical activity features). The training and testing data used in this work are drawn from an officially released subset of the ToN IoT dataset, which includes 300,000 normal traffic flows and 20,000 flows for each attack category, except for the XSS attack class, which contains only 1,043 recorded flows. We rely on the IoT-telemetry splits of the ToN-IoT corpus, where each physical sensor is provided as an independent CSV file containing pre-divided train/test records across the security classes (i.e., benign or attack type), which we have merged into a single dataset for comprehensive analysis. The same dataset was processed in several steps to prepare it for ML model training. This section describes each pre-processing step, including handling missing data, feature normalization, and categorical data encoding.

Algorithm 1 MI–SMOTE–Boost intrusion detection

```
Require: Dataset $D$, top-$k$, $k_{\text{smote}}$
Ensure: Trained models $\mathcal{M}$
 1: Encode, impute, scale $D$
 2: Compute MI; select top-$k$ features $\mathbf{X}_k$
 3: Split $D \to D_{\text{train}}, D_{\text{test}}$
 4: Apply SMOTE($k_{\text{smote}}$) on $D_{\text{train}}[\mathbf{X}_k]$
 5: for each booster $\in$ {XGBoost, LightGBM, ADABoost, CATBoost} do
 6:     Train booster on balanced $D_{\text{train}}[\mathbf{X}_k]$
 7:     Evaluate on $D_{\text{test}}[\mathbf{X}_k]$
 8:     Store metrics $\to \mathcal{M}$
 9: end for
10: return $\mathcal{M}$
```

3.1.1. Handling missing data

Handling missing values is an important step in data cleaning, and it is crucial for ensuring the integrity and completeness of the dataset. For numeric features with missing values, mean imputation was applied. This involves replacing missing values in a feature x_j with the mean of that feature computed over the numerical data subset. This approach ensures that all data records can be used for training without introducing significant bias. The imputation formula used is:

$$x_{ij} \leftarrow \frac{1}{|D_{\text{numeric}}|} \sum_{k \in D_{\text{numeric}}} x_{kj} \quad \text{if} \quad x_{ij} \text{ is missing} \tag{1}$$

Where D_{numeric} represents the numerical data subset, x_{ij} is the missing value, and the mean of the feature x_j is computed over the entire numerical data subset. For categorical features, missing values were handled separately. In the ToN-IoT dataset, any missing categories were imputed using the most frequent value (i.e., mode) within the data, ensuring that the categories are consistent across the dataset.

3.1.2. Feature normalization

To guarantee that every numerical feature plays an active role in the model training, standardization was applied using the $StandardScaler$ from scikit-learn. This transformation ensures that each feature has a mean of zero and a standard deviation of one, which prevents features with larger numeric ranges from disproportionately influencing the model. The standardization formula used is:

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \tag{2}$$

Where x'_{ij} is the standardized value of feature x_j for the i-th instance, μ_j is the mean, and σ_j is the standard deviation for feature x_j across the data. This scaling was applied to training and testing sets, ensuring that all features are treated consistently across both training and testing phases.

3.1.3. Categorical data encoding

Categorical features, such as traffic type or device identifiers, were transformed into numerical representations using one-hot encoding. This process creates binary columns for each unique category in a feature. For example, if a feature protocol has three unique values (e.g., TCP, UDP, and $ICMP$), one-hot encoding would generate three binary columns: $protocol\_TCP$, $protocol\_UDP$, $protocol\_ICMP$. Each instance of the dataset is represented by 1 in the corresponding category column and 0 in the others. This transformation prevents the model from assuming any ordinal relationship between categories and ensures that categorical variables are processed appropriately by tree-based ML models.

3.2. Feature engineering

MI quantifies the amount of information one variable provides about another [28]. In classification tasks, MI is used to select features that have the highest dependency on the class labels. To mitigate the issue of dimensionality, MI was used to evaluate the relevance of each feature X_j with respect to the multiclass label Y. The MI score quantifies the reduction in label uncertainty due to knowledge of the feature, with the formula:

$$I(X_j; Y) = \sum_{x_j} \sum_{y} p(x_j, y) \log \frac{p(x_j, y)}{p(x_j)p(y)} \tag{3}$$

Where $p(x_j, y)$ is the joint probability of feature X_j and label Y, and $p(x_j)$, $p(y)$ are the marginal probabilities of X_j and Y, respectively. The MI score measures the amount of information shared between a feature and the target, with higher values indicating stronger relevance.

In this study, we applied the $SelectKBest$ method from scikit-learn, using $mutual\_info\_classif()$ as the scoring function to select the top $k = 10$ features that exhibit the highest MI with the target label. This selection helps to eliminate irrelevant, redundant, or noisy features, improving the performance of the model by reducing overfitting and making the learning process more efficient. The selected features were used for model training, ensuring that only the most informative predictors were considered.

3.3. Class balancing

Imbalanced class distributions constitute a critical challenge in intrusion detection, particularly in IIoT environments where normal traffic vastly outweighs malicious instances. This imbalance leads to biased decision boundaries that favor the majority class, resulting in high false-negative rates for minority class predictions. SMOTE was applied to address this issue by generating synthetic instances for the minority class through interpolation [29].

SMOTE synthesizes new instances by sampling from the minority class x and selecting one of its k_{nn} nearest neighbors, x_{nn}. A synthetic example is created by adding a scaled difference between the minority sample and its neighbor:

$$\tilde{x} = x + \lambda(x_{\text{nn}} - x), \qquad \lambda \sim \mathcal{U}(0, 1) \tag{4}$$

Where λ is a randomly chosen value between 0 and 1, ensuring that the synthetic instance lies somewhere between x and x_{nn} in the feature space. This process is repeated for all minority instances until class distribution approaches balance, effectively enlarging the minority class manifold and promoting wider decision margins. This re-balancing approach helps improve the model's ability to learn from both minority and majority classes equally, reducing the occurrence of false negatives and false positives during model prediction.

3.4. Gradient-boosted model development

3.4.1. Data partitioning

The data partitioning stage is essential in ML pipeline. After pre-processing, the dataset was randomly split into 80% for training and 20% for testing, which is a standard approach in ML for model evaluation. Additionally, 10-fold cross-validation was employed to assess each model's performance and generalizability. This technique involves splitting the dataset into ten equal partitions. Each fold serves as a test set once, while the remaining nine folds are used for training. This process ensures that every data point is utilized for both training and testing. By employing this strategy, overfitting is minimized, and a more accurate estimate of the model's performance is obtained compared to a single train-test split.

3.4.2. Model fitting and validation

In this study, four powerful ensemble learners, such as XGBoost, LightGBM, AdaBoost, and CatBoost, were independently trained to evaluate their effectiveness in detecting intrusions in IIoT environments. These models were chosen for their capacity to efficiently process large-scale datasets, capture complex feature patterns, and provide high accuracy with relatively fast training times [30]. The models were trained using the balanced, ten-feature design matrix. Each model's objective function and working mechanism are described in detail, focusing on how they iteratively improve their performance during the training process.

- XGBoost: it provides a highly efficient, scalable form of gradient boosting by sequentially constructing decision trees, each trained to rectify the residual errors of the preceding ensemble, and minimizes a regularized additive loss function:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{N} \ell\big(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\big) + \Omega(f_t) \tag{5}$$

  Where N is the number of samples, ℓ is the loss function, usually multinomial logistic loss for classification tasks, y_i is the true label of the i-th sample, $\hat{y}_i^{(t-1)}$ is the prediction from the previous iteration, $f_t(x_i)$ is the decision function of the tree at iteration t for sample x_i, and $\Omega(f_t)$ is the regularization term that penalizes the complexity of the decision tree f_t. The regularization term $\Omega(f_t)$ helps prevent overfitting by controlling the complexity of the model. It is defined as:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{6}$$

  Where T is the number of leaves in the tree, w_j represents the weight of the j-th leaf, and γ and λ are hyperparameters controlling the complexity of the tree. The goal of XGBoost is to minimize this objective function by balancing model fit to data while preventing overfitting by penalizing large trees.
- LightGBM: it is a gradient boosting framework designed to handle large datasets with higher efficiency than traditional gradient boosting methods like XGBoost. Similar to XGBoost, LightGBM builds an ensemble of trees sequentially, with each tree focusing on the residuals of the previous one. Its objective function is defined similarly to that of XGBoost, with an additional focus on efficiency and speed. The regularization term in LightGBM is given by:

$$\Omega(f_t) = \lambda \sum_{j=1}^{T} w_j^2 \tag{7}$$

  Where λ is a regularization parameter, and w_j represents the weight of the j-th leaf. Additionally, LightGBM employs a histogram-based approach for training, which speeds up computation by approximating the feature values into discrete bins, reducing the computational cost of finding the best split for each feature.
- AdaBoost: it is an ensemble technique that forms a robust model by aggregating weak learners and iteratively up-weighting the misclassified samples, forcing the model to focus more on hard-to-classify examples in subsequent iterations. AdaBoost iteratively adjusts the weight of each weak classifier, and the final prediction is the weighted sum of all weak classifiers. It minimizes the weighted error by adjusting

the weights of the training instances after each iteration. The weight update rule for the i-th sample in AdaBoost is:

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{8}$$

Where ϵ_t is the weighted error of the weak learner in the t-th iteration. The final prediction is obtained by combining the weak learners using their weights, where the weak learners with lower errors are given higher weights:

$$f(x) = \text{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right) \tag{9}$$

Where $h_t(x)$ is the weak classifier at iteration t, α_t is the weight assigned to the weak classifier at iteration t, and $f(x)$ is the final prediction, which is the weighted sum of the weak classifiers' predictions.

– CatBoost: it is specifically developed to process categorical variables with high efficiency by converting them into numerical representations via an efficient algorithm that accounts for the order of categories and prevents target leakage, which is beneficial in IIoT environments where categorical data, such as device types or protocols, are common. CatBoost applies an 'ordered boosting' approach, which mitigates target leakage and prevents overfitting. The objective function for CatBoost is similar to that of XGBoost and LightGBM, with an additional emphasis on categorical feature handling. The regularization term controls the complexity of the trees and prevents overfitting.

3.5. Evaluation strategy

This work employs a suite of evaluation metrics—F1-score, accuracy, precision, FPR, and recall—to rigorously quantify the effectiveness of IIoT-oriented IDS. These metrics collectively provide a nuanced view of classification performance. This perspective is especially critical when addressing the class imbalance characteristic of intrusion detection datasets.

4. RESULTS AND DISCUSSION

This section presents the experimental findings, comparative analysis, and a comprehensive discussion regarding the performance improvements achieved through the proposed technique.

4.1. Experimental setup

All experiments were conducted on the current version of Google Colab, operating in a cloud environment equipped with dual Intel® Xeon® virtual CPUs, approximately 12 GB of system memory. Programming was performed in Python 3.10, utilizing standard ML and data processing libraries, including scikit-learn, imbalanced-learn, XGBoost, LightGBM, and CatBoost, alongside visualization tools such as matplotlib and seaborn. Each classifier—AdaBoost, CatBoost, LightGBM, and XGBoost—was trained initially on the raw feature set (i.e., baseline) and subsequently on a feature subset selected via MI, with class imbalance addressed through SMOTE. Model evaluation employed 10-fold stratified cross-validation and captured key performance indicators: F1-score, precision, accuracy, recall, FPR, training time, and prediction time, enabling a comprehensive comparison of model behavior before and after feature engineering and class balancing.

4.2. Global performance comparison

The global performance comparison across all models is summarized in Table 1. Each model was evaluated in two scenarios: baseline (i.e., before MI-SMOTE) and enhanced (i.e., after MI-SMOTE). As illustrated in Table 1, all models achieved substantial improvements across key performance indicators after the application of MI-SMOTE. Accuracy converged to approximately 99.43% across all models. Notably, AdaBoost, which initially had the lowest performance, exhibited the greatest relative improvement in both classification metrics and computational efficiency. The F1-score, which balances precision and recall, is a key indicator of classification.

Table 1. Performance comparison of ensemble models before and after MI-SMOTE

Metric	LightGBM	XGBoost	CatBoost	AdaBoost
Accuracy (before)	0.9930	0.9943	0.9916	0.9859
Accuracy (after)	0.9943	0.9943	0.9943	0.9943
Precision (before)	0.9931	0.9940	0.9920	0.9866
Precision (after)	0.9945	0.9945	0.9945	0.9945
Recall (before)	0.9930	0.9943	0.9916	0.9859
Recall (after)	0.9943	0.9943	0.9943	0.9943
F1-score (before)	0.9930	0.9937	0.9907	0.9851
F1-score (after)	0.9937	0.9939	0.9937	0.9937
FPR (before)	0.00094	0.00082	0.00118	0.00198
FPR (after)	0.00080	0.00080	0.00080	0.00080
Training time (before) (s)	7.5607	12.1173	10.5279	91.8730
Training time (after) (s)	4.0427	3.6500	7.2592	15.2537
Prediction time (before) (s)	0.9968	0.3159	0.4913	6.9909
Prediction time (after) (s)	0.3893	0.1745	0.1430	2.7225
Total time (before) (s)	8.5575	12.4332	11.0192	98.8639
Total time (after) (s)	4.4320	3.8245	7.4022	17.9762

As shown in Figure 2, all models demonstrated an increase in F1-score after the application of MI-SMOTE. AdaBoost experienced the most significant improvement, increasing from 98.51% to 99.37%, while CatBoost improved from 99.07% to 99.37%. LightGBM and XGBoost also showed slight but consistent gains, stabilizing near 99.37% and 99.39%. Minimizing FPR is crucial, especially in applications where false alarms carry high costs. The FPR evolution is presented in Figure 3, where it shows a consistent reduction in FPR across all models. Initially, AdaBoost and CatBoost exhibited higher FPR values of 0.00198 and 0.00118, respectively. After applying MI-SMOTE, all models achieved a reduced and unified FPR of 0.00080.

Efficiency in terms of computational resources is another critical aspect for real-world deployment. The impact of MI-SMOTE on training and prediction times is depicted in Figure 4, illustrating that training and prediction times were generally reduced after pre-processing, feature engineering, and class balancing phases. AdaBoost benefited significantly, reducing its total execution time from approximately 99 seconds to 18 seconds. XGBoost and LightGBM also achieved considerable reductions in training and prediction times, confirming the efficiency gain of the approach's steps. CatBoost, after MI-SMOTE, managed to decrease its total time to approximately 7.4 seconds. Overall, the integration of MI-based feature selection and SMOTE-based balancing substantially enhanced classification robustness, minimized false alarms, and optimized computational efficiency across all evaluated models.

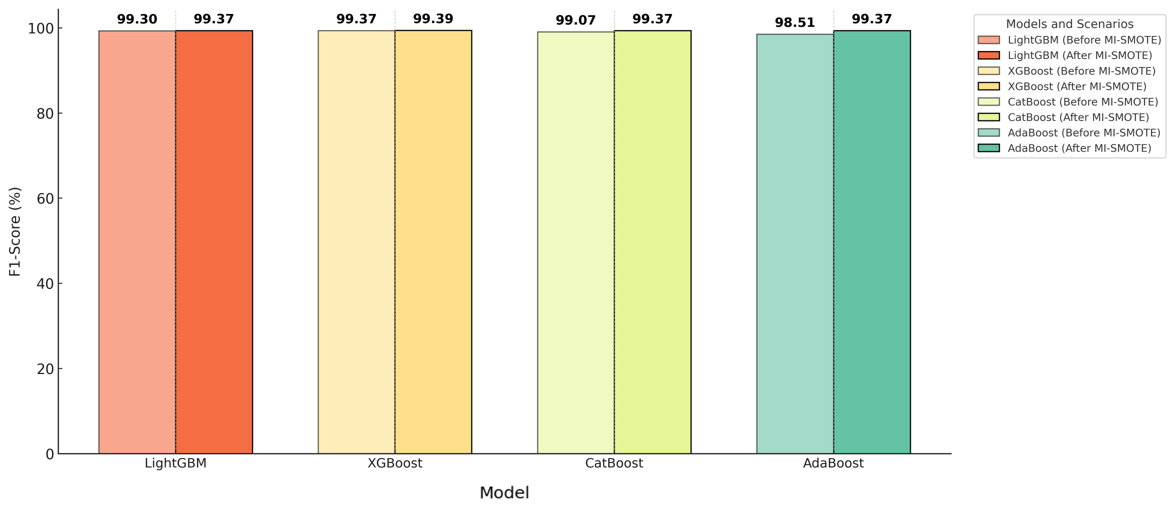

Figure 2. Comparison of F1-scores for all models before and after MI-SMOTE application

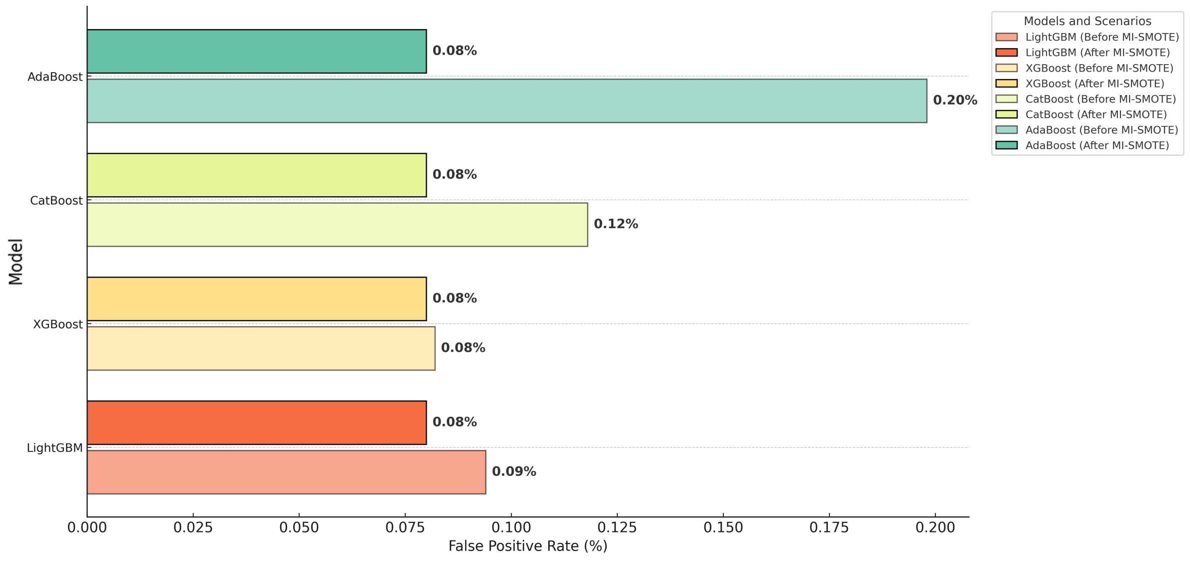

Figure 3. FPR for all models before and after MI-SMOTE integration decreases consistently to 0.080%

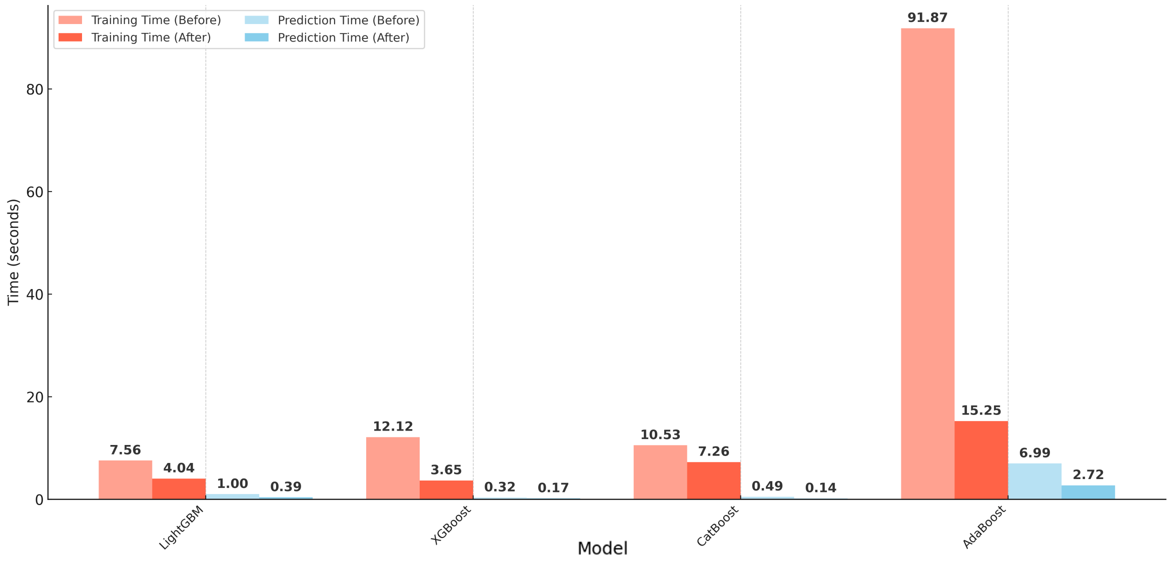

Figure 4. Execution time analysis showing training and prediction durations before and after MI-SMOTE

5. CONCLUSION

This study aimed to investigate the impact of integrating MI feature selection and SMOTE class balancing techniques on the performance of ensemble learning models for classification tasks. As initially stated, the objective was to enhance predictive accuracy, reduce FPR, and optimize computational efficiency. The experimental results confirm that these objectives were successfully achieved. All evaluated models, such as LightGBM, XGBoost, CatBoost, and AdaBoost, showed consistent improvements in classification metrics after the application of MI-SMOTE. Notably, F1-scores exceeded 99.37% across all models, while FPR was uniformly reduced to 0.080%. Additionally, significant reductions in training and prediction times were observed for several models, further validating the effectiveness of the framework's stages. These findings not only demonstrate the compatibility between the research objectives and outcomes but also highlight the practicality of the proposed approach for real-world large-scale deployments where both performance and efficiency are critical. Prospects for future work include the extension of this methodology to more diverse and imbalanced IIoT datasets (e.g., NF-ToN-IoT-v2, UNSW-NB15) to assess generalizability across different environments. Additionally, we plan to conduct an ablation study to isolate and analyze the individual impacts of MI-based feature selection and SMOTE balancing techniques on classification performance. The exploration of adaptive or dynamic feature selection strategies beyond MI, such as hybrid filter-wrapper methods, and the integration of balancing approaches that are dynamically tailored to the nature of specific attack categories represent promising enhancements. Furthermore, we intend to explore explainable artificial intelligence (XAI) tools such as Shapley additive explanations (SHAP) and local interpretable model-agnostic explanations (LIME) to improve the interpretability of model decisions and support transparency in real-world deployments. Finally, investigating zero-day threat detection capabilities using anomaly-based learning or few-shot learning models will also be considered to bolster resilience against unknown attacks.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	**C**	**M**	**So**	**Va**	**Fo**	**I**	**R**	**D**	**O**	**E**	**Vi**	**Su**	**P**	**Fu**
Mouad Choukhairi	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	
Oumaima Chentoufi	✓	✓			✓	✓		✓		✓				
Ouail Choukhairi	✓	✓			✓	✓		✓		✓				
Youssef Fakhri		✓		✓	✓		✓			✓		✓	✓	

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject Administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding Acquisition
Fo	: **Fo**rmal Analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

This study did not involve human participants, and informed consent was therefore not required.

ETHICAL APPROVAL

This research did not involve human or animal subjects and did not require ethical approval.

DATA AVAILABILITY

The data that supports the findings of this study is openly available in The ToN-IoT dataset at: https://research.unsw.edu.au/projects/toniot-datasets.

REFERENCES

[1] S. H. Jafier, "Utilizing feature selection techniques in intrusion detection system for internet of things," in *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems*, 2018, pp. 1–3, doi: 10.1145/3231053.3234323.

[2] A. M. Vulfin, V. I. Vasilyev, V. E. Gvozdev, K. V. Mironov, and O. E. Churkin, "Network traffic analysis based on machine learning methods," *Journal of Physics: Conference Series*, vol. 2001, no. 1, 2021, doi: 10.1088/1742-6596/2001/1/012017.

[3] L. Santos, R. Gonçalves, C. Rabadão, and J. Martins, "A flow-based intrusion detection framework for internet of things networks," *Cluster Computing*, vol. 26, no. 1, pp. 37–57, 2023, doi: 10.1007/s10586-021-03238-y.

[4] J. R. Rose, M. Swann, G. Bendiab, S. Shiaeles, and N. Kolokotronis, "Intrusion detection using network traffic profiling and machine learning for IoT," in *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, 2021, pp. 409–415, doi: 10.1109/NetSoft51509.2021.9492685.

[5] M. Choukhairi, S. Tahiri, O. Choukhairi, Y. Fakhri, and M. Amnai, "Butterfly optimization-based ensemble learning strategy for advanced intrusion detection in internet of things networks," *International Journal of Electrical and Computer Engineering*, vol. 15, no. 3, pp. 3494–3505, 2025, doi: 10.11591/ijece.v15i3.pp3494-3505.

[6] P. Kumar and A. K. M. N. Islam, "Interpretable cyber threat detection for enterprise industrial networks: a computational design science approach," *arxiv.2409.03798*, 2024.

[7] G. Thamilarasu and S. Chawla, "Towards deep-learning-driven intrusion detection for the internet of things," *Sensors*, vol. 19, no. 9, 2019, doi: 10.3390/s19091977.

[8] S. F. Chevtchenko *et al.*, "Anomaly detection in industrial machinery using IoT devices and machine learning: a systematic mapping," *IEEE Access*, vol. 11, pp. 128288–128305, 2023, doi: 10.1109/ACCESS.2023.3333242.

[9] T. Mazhar *et al.*, "Analysis of IoT security challenges and its solutions using artificial intelligence," *Brain Sciences*, vol. 13, no. 4, 2023, doi: 10.3390/brainsci13040683.

[10] P. Vanin *et al.*, "A study of network intrusion detection systems using artificial intelligence/machine learning," *Applied Sciences*, vol. 12, no. 22, 2022, doi: 10.3390/app122211752.

[11] S. Msika, A. Quintero, and F. Khomh, "SIGMA: Strengthening IDS with GAN and metaheuristics attacks," in *ICIMP 2021: The Sixteenth International Conference on Internet Monitoring and Protection*, 2019, pp. 10–20.

[12] Y. Meidan, D. Avraham, H. Libhaber, and A. Shabtai, "CADeSH: collaborative anomaly detection for smart homes," *IEEE Internet of Things Journal*, vol. 10, no. 10, pp. 8514–8532, 2023, doi: 10.1109/JIOT.2022.3194813.

[13] A. H. K. Mohammed, H. Jebamikyous, D. Nawara, and R. Kashef, "IoT cyber-attack detection: a comparative analysis," in *International Conference on Data Science, E-learning and Information Systems 2021*, 2021, pp. 117–123, doi: 10.1145/3460620.3460742.

[14] F. Amiri, M. M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011, doi: 10.1016/j.jnca.2011.01.002.

[15] R.-H. Dong, D.-F. Wu, Q.-Y. Zhang, and H. Duan, "Mutual information-based intrusion detection model for industrial internet," *International Journal of Network Security*, vol. 20, no. 1, pp. 131–140, 2018, doi: 10.6633/IJNS.201801.20(1).14.

[16] Sharipuddin *et al.*, "Features extraction on IoT intrusion detection system using principal components analysis (PCA)," in *2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics (EECSI)*, 2020, pp. 114–118, doi: 10.23919/EECSI50503.2020.9251292.

[17] C. Oumaima, C. Mouad, C. Khalid, and A. Ilyas, "Exploring the impact of PCA variants on intrusion detection system performance," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 5, pp. 392–400, 2024, doi: 10.14569/IJACSA.2024.0150539.

[18] R. H. Mohamed, F. A. Mosa, and R. A. Sadek, "Efficient intrusion detection system for IoT environment," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 4, 2022, doi: 10.14569/IJACSA.2022.0130467.

[19] A. Ghubaish, Z. Yang, A. Erbad, and R. Jain, "LEMDA: a novel feature engineering method for intrusion detection in IoT systems," *IEEE Internet of Things Journal*, vol. 11, no. 8, pp. 13247–13256, 2024, doi: 10.1109/JIOT.2023.3328795.

[20] R. Singh and R. L. Ujjwal, "Feature selection methods for IoT intrusion detection system: comparative study," in *Computational Intelligence: Select Proceedings of InCITe 2022*, Singapore: Springer, 2023, pp. 227–236, doi: 10.1007/978-981-19-7346-8_20.

[21] A. Houkan *et al.*, "Enhancing security in industrial IoT networks: machine learning solutions for feature selection and reduction," *IEEE Access*, vol. 12, pp. 160864–160883, 2024, doi: 10.1109/ACCESS.2024.3481459.

[22] H. Saputro, T. Ahmad, and M. A. R. Putra, "Enhancing IoT Botnet detection: a comparative study of balancing techniques with naive Bayes classifier," in *2024 International Conference on Information Technology and Computing (ICITCOM)*, 2024, pp. 260–265, doi: 10.1109/ICITCOM62788.2024.10762450.

[23] S. S. M. Than, A. M. Soe, and A. H. Maw, "Investigation of oversampling in IoT-IDS," in *2024 IEEE Conference on Computer Applications (ICCA)*, 2024, pp. 1–6, doi: 10.1109/ICCA62361.2024.10533055.

[24] Z. Saharuna and T. Ahmad, "Multiclass imbalance resampling techniques for network intrusion detection," in *2024 10th International Conference on Smart Computing and Communication (ICSCC)*, 2024, pp. 450–454, doi: 10.1109/ICSCC62041.2024.10690582.

[25] T.-T.-H. Le, Y. E. Oktian, and H. Kim, "XGBoost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems," *Sustainability*, vol. 14, no. 14, 2022, doi: 10.3390/su14148707.

[26] J. Al Amien, H. Ab Ghani, N. Izrin Md Saleh, S. Soni, Y. Fatma, and R. Hayami, "A comprehensive evaluation of multiclass imbalance techniques with ensemble models in IoT environments," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 22, no. 3, pp. 690–701, 2024, doi: 10.12928/telkomnika.v22i3.25887.

[27] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_IoT telemetry dataset: a new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020, doi: 10.1109/ACCESS.2020.3022862.

[28] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005, doi: 10.1109/TPAMI.2005.159.

[29] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique nitesh," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002, doi: 10.1613/jair.953.

[30] M. Choukhairi, M. M'Haouach, Y. Fakhri, and M. Amnai, "Tree-driven intelligent security system for intrusion detection in IoT environment," in *2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2023, pp. 1–7, doi: 10.1109/WINCOM59760.2023.10322954.

BIOGRAPHIES OF AUTHORS

Mouad Choukhairi received his bachelor's degree (B.Sc.) in 2018 at Mathematics and Computer Science and his master's degree in Big Data & Cloud Computing in 2020 from the Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco. He is currently a Ph.D. student in the Computer Science Research Laboratory (LaRI) in the field of cybersecurity, IDS, big data, ML, DL, and IoT. He can be contacted at email: mouad.choukhairi@uit.ac.ma.

Oumaima Chentoufi obtained an engineering degree in Telecommunications and Network Engineering in 2019 from the National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco. She is presently a Ph.D. student at the Engineering Science Laboratory within the same institution, where her research interests encompass cybersecurity, intrusion detection systems, and artificial intelligence. She can be contacted at email: oumaima.chentoufi@uit.ac.ma.

Ouail Choukhairi received his bachelor's degree (B.Sc.) in 2020 at Mathematics and Computer Science and his master's degree in Big Data & Cloud Computing in 2022 from Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco. He is currently a Ph.D. student in the Computer Science Research Laboratory (LaRI) in the field of artificial intelligence, semantic segmentation, ML, and DL. He can be contacted at email: ouail.choukhairi@uit.ac.ma.

Youssef Fakhri obtained a Bachelor of Science degree in Electronics and a Diploma of Advanced Scientific Studies (DESA) in Computer Science and Telecommunications from the Faculty of Sciences of Rabat (Mohammed V University - Agdal, Rabat, Morocco), in 2001 and 2003 respectively. He obtained a Ph.D. thesis on November 17, 2007, from the University Mohammed V - Agdal, Rabat, Morocco in collaboration with the Polytechnic University of Catalonia (UPC), Spain. His research topics are information theory, signal processing, wireless telecommunications (WSN), and routing protocols. He is now a Professor of Higher Education in Computer Science at the Faculty of Sciences of Kenitra. He is a head of Research Laboratory in Computer Science (LaRI). Further info on his homepage: https://sites.google.com/site/proffakhri/. He can be contacted at email: fakhri@uit.ac.ma.