



ARTICLE

A Robustness Aware Intrusion Detection Framework Using Ensemble Learning and Imbalance Aware Preprocessing for Smart Home IoT Networks

Adi Wibowo^{1,2}, Deris Stiawan^{3,*}, Chonlameth Arpnikanondt^{4,*}, Ahmad Heryanto³, Didi Rosiyadi^{5,*}, Mohd Yazid Idris⁶, Fandi Kurniawan², Tuul Triyason⁴ and Rahmat Budiarto⁷

¹Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya, Palembang, Indonesia

²Faculty of Engineering and Computer Science, Universitas Muhammadiyah, Kotabumi, Indonesia

³Department of Computer Engineering, Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia

⁴School of Information Technology, King Mongkut's University of Technology Thonburi, Bangkok, Thailand

⁵Research Center for Artificial Intelligence and Cybersecurity, National Research and Innovation Agency, Jakarta, Indonesia

⁶Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru, Johor, Malaysia

⁷College of Computing and Information, Al-Baha University, Alaqiq, Saudi Arabia

*Corresponding Authors: Deris Stiawan. Email: deris@unsri.ac.id; Chonlameth Arpnikanondt. Email: chonlameth@sit.kmutt.ac.th; Didi Rosiyadi. Email: didi016@brin.go.id

Received: 07 May 2026; Accepted: 22 July 2026; Published: 22 October 2026

ABSTRACT: The proliferation of smart home Internet of Things (IoT) devices has introduced critical security vulnerabilities that demand robust and efficient intrusion detection mechanisms. Existing machine learning-based Intrusion Detection Systems (IDS) often suffer from class imbalance bias, limited cross domain generalization, and insufficient robustness evaluation under adversarial conditions. This study proposes a robustness-aware intrusion detection framework that systematically integrates imbalance-aware preprocessing and ensemble machine learning into a unified pipeline specifically designed for resource-constrained smart home IoT environments. The framework employs Min Max normalization and Synthetic Minority Over sampling Technique (SMOTE) with strict train-test separation to prevent data leakage, operating directly on the seven packet-level traffic features available in the capture rather than on a reduced subset. Three machine learning models Decision Tree (DT), Random Forest (RF), and XGBoost are systematically compared against deep learning baselines (CNN and LSTM) across three benchmark IoT datasets: COMNETS Smart Home, IoT-23, and TON IoT. Under a conventional random-split evaluation, XGBoost achieves the highest detection performance (97.8% accuracy, 97.0% F1) at substantially lower computational cost than the deep-learning alternatives; a leakage-controlled evaluation, however, shows that much of this reflects evaluation artifacts, with the honest macro-F1 falling to 0.558 and reliable detection of the scarce benign (minority) class remaining unverified. Cross-dataset experiments are reported as a preliminary analysis; because the feature spaces could only be approximately harmonized, these results are indicative rather than conclusive evidence of cross-domain generalization. Robustness analysis under Gaussian noise injection (5%–15%) and random feature perturbation (not gradient-based adversarial attacks) reveals that ensemble models exhibit significantly lower performance degradation compared to single-tree approaches. Statistical significance testing (paired *t*-test, $p < 0.05$) validates that the observed improvements are not attributable to random variation. These results show that, once evaluation artifacts are controlled for, the honest performance of these lightweight models is modest, underscoring the importance of leakage-aware evaluation for smart-home IoT intrusion detection.

KEYWORDS: Intrusion detection system; IoT security; ensemble learning; robustness analysis; SMOTE; smart home; cross-dataset validation

1 Introduction

The rapid proliferation of IoT technologies in smart home environments has significantly transformed modern living by enabling automation, improving operational efficiency, and enhancing user convenience. Smart home ecosystems consist of interconnected devices, such as surveillance cameras, smart locks, and environmental sensors, which continuously exchange data to support real-time monitoring and intelligent control. However, this increasing interconnectivity also introduces critical security challenges, as many IoT devices operate under resource constraints and lack robust built-in protection mechanisms, making them highly vulnerable to cyber threats [1,2]. Consequently, smart home networks have become attractive targets for various cyber-attacks, including Distributed Denial of Service (DDoS), Man-in-the-Middle (MITM), and botnet-based attacks, which can significantly disrupt system functionality and compromise user privacy [3,4]. To mitigate these risks, Intrusion Detection Systems (IDS) have been widely adopted as an additional security layer to monitor network behavior and detect anomalous activities in real time [5].

Traditional signature based IDS approaches are effective in identifying known attack patterns; however, they struggle to detect previously unseen or zero-day attacks due to their reliance on predefined rules and static signature databases [5]. To overcome these limitations, machine learning-based IDS approaches have gained significant attention due to their ability to automatically learn complex patterns from network traffic data and adapt to evolving attack behaviors [6,7]. Despite these advancements, several critical challenges remain. First, many existing studies rely on generic or benchmark datasets that do not adequately capture the unique characteristics of smart home IoT traffic, thereby limiting model generalization in real-world environments [8,9]. Second, class imbalance is often insufficiently addressed, resulting in biased models that favor dominant classes while failing to detect minority yet critical attack patterns [8,10]. Third, computational efficiency and deployment feasibility are frequently overlooked, even though they are essential in resource-constrained IoT environments [11,12]. Finally, robustness under noisy or adversarial conditions is rarely evaluated, leading to overly optimistic performance assessments and reduced reliability in real-world deployments [13,14].

To address these challenges, this study proposes a robustness-aware intrusion detection framework specifically designed for smart home IoT environments. The proposed framework integrates imbalance-aware preprocessing and machine learning-based classification into a unified and efficient pipeline. The framework is also evaluated under a leakage-controlled protocol to confirm that the reported performance reflects genuine detection ability rather than evaluation artifacts. This study is guided by three main research questions. The first question investigates which machine learning algorithm provides the most effective and stable performance for intrusion detection in smart home IoT environments. The second question examines the extent to which preprocessing improves detection under imbalanced traffic conditions. The third question asks how detection performance changes under a leakage-controlled, temporally ordered evaluation. The main contributions of this study are summarized as follows:

1. A robustness-aware intrusion detection framework is developed by integrating preprocessing techniques and ensemble-based machine learning into a unified pipeline tailored for resource-constrained IoT systems.
2. The impact of preprocessing techniques, particularly SMOTE-based imbalance handling and normalization, is systematically analyzed to improve detection performance under imbalanced data conditions [15].
3. A leakage-controlled validity analysis is conducted to quantify how duplicate records, an absolute-time feature, and random splitting affect the reported performance, complementing the standard evaluation with honest baselines.

2 Related Work

The rapid development of IoT technologies has significantly increased the adoption of smart home systems, enabling seamless communication among heterogeneous devices. However, this interconnectivity also expands the attack surface, making IoT environments highly vulnerable to cyber threats such as Distributed Denial of Service (DDoS), Man-in-the-Middle (MITM), and botnet-based attacks [4,16]. Due to the inherent characteristics of IoT systems, including limited computational resources and dynamic network behavior, ensuring security in such environments remains a critical challenge. IDSs have been widely investigated as a key solution for monitoring network activities and identifying malicious behavior. Traditional IDS approaches, including signature-based and anomaly-based methods, have shown certain limitations, particularly in detecting zero-day attacks and maintaining low false positive rates in dynamic environments [13,17]. To address these issues, machine learning (ML) and deep learning (DL) techniques have been increasingly adopted to enhance detection capability and adaptability [11,16]. Table 1 presents a comparative analysis of existing studies on IDS in IoT environments, highlighting their strengths, limitations, and research gaps.

Table 1: Comparative analysis of existing IDS approaches.

Study	Approach	Strength	Limitation	Research Gap
Ahmad & Alsmadi [6]	ML-based IDS	Adaptive to traffic patterns	Limited experimental validation	Lack of real-world IoT evaluation
Kantharaju et al. [7]	ML Framework	High detection	Limited generalization	No cross-dataset validation
Ullah et al. [17]	DL (Transformer)	Captures complex patterns	High computational cost	Not suitable for IoT devices
Ibrahim Hairab et al. [18]	DL-based IDS	Strong detection capability	Resource-intensive	Limited deployment feasibility
Kivanç Eren et al. [12]	CNN-based IDS	Effective for zero-day attacks	High latency	Not real-time applicable
Chimphlee and Chimphlee [19]	XGBoost	High accuracy & efficiency	Dataset dependency	Limited robustness analysis
Guezzaz et al. [16]	Decision Tree	Interpretable model	Overfitting issue	Weak generalization
Hamidou and Mehdi [20]	Random Forest	Robust performance	High memory usage	Not lightweight
Rezvani & Wang [9]	Imbalance Learning	Handles class imbalance	Not IDS-specific	Lack of integration into IDS
Berhili et al. [8]	Survey	Comprehensive overview	No implementation	Lack of experimental validation
This Study	ML (DT, RF, XGBoost)	Balanced performance	–	Lightweight + robust + multi-evaluation

As shown in Table 1, machine learning and deep learning-based IDS approaches have demonstrated strong detection performance across various IoT scenarios. Deep learning models, such as CNN and transformer-based architectures, are particularly effective in capturing complex patterns in network traffic

data. However, their high computational complexity and resource requirements limit their applicability in real-time and resource-constrained IoT environments [19]. On the other hand, classical machine learning methods, including Decision Tree, Random Forest, and XGBoost, offer a better balance between performance and computational efficiency. Nevertheless, these approaches still present certain limitations. Decision Tree models tend to overfit in high-dimensional datasets, while ensemble methods such as Random Forest and XGBoost, although more robust, require higher memory and computational resources [16,20]. Another critical issue identified from previous studies is the class imbalance problem, where normal traffic dominates malicious traffic. This imbalance leads to biased models that struggle to detect minority attack classes [8]. Although techniques such as SMOTE have been proposed to address this issue, their integration into IDS frameworks remains limited and often treated as a secondary preprocessing step rather than a core component of the detection pipeline [15].

Furthermore, most existing studies rely on single datasets, which limits the generalization capability of the proposed models across different IoT environments [21]. In addition, robustness under noisy or adversarial conditions is rarely evaluated, resulting in models that perform well in controlled settings but fail in real-world scenarios [13]. Based on these observations, it can be concluded that there is a need for a comprehensive IDS framework that integrates imbalance-aware preprocessing, efficient machine learning models, and robust evaluation strategies. This study addresses these challenges by proposing a lightweight and robustness-aware intrusion detection framework designed specifically for smart home IoT environments, aiming to achieve an optimal balance between detection performance, computational efficiency, and generalization capability [19].

Data Leakage and Evaluation Validity in ML-Based IDS

A growing body of work shows that reported performance of machine-learning IDS is often inflated by methodological issues rather than genuine detection ability. Sommer and Paxson [21] noted the gap between research results and operational reality, Ref. [22] provided a pragmatic assessment of ML for network intrusion detection and catalogued common pitfalls such as data snooping and inappropriate metrics, and Ref. [23] linked data leakage to a broader reproducibility crisis in machine-learning-based science. Dataset quality is also a concern: Engelen et al. [24] found construction and labelling errors in a widely used dataset, Ref. [25] surveyed NIDS dataset limitations, Ref. [26] benchmarked ML models under controlled conditions, Ref. [14] stressed realistic adversarial evaluation. This study contributes a leakage-controlled evaluation on a smart-home capture.

Recent directions in IoT intrusion detection also include federated-learning IDS, explainable (XAI) IDS, transformer-based detectors, and AutoML pipeline optimization.

3 Methodology

To support reproducibility, the full configuration is reported here. Experiments were run in Python 3.12 with scikit-learn 1.6, XGBoost 3.2, and imbalanced-learn 0.14 on a standard Google Colab CPU runtime. Hyperparameters: Decision Tree (max depth 20); Random Forest (60 trees, max depth 20); XGBoost (300 trees, max depth 8, learning rate 0.1, subsample and colsample 0.9, histogram tree method). Min-Max normalization and SMOTE were fitted on the training partition only and, within cross-validation, on the training folds only never on test data.

3.1 System Overview

In this study, we propose a robustness-aware intrusion detection framework tailored for smart home Internet of Things (IoT) environments, where security challenges are intensified by constrained computational resources, heterogeneous communication protocols, and dynamic network behaviors [4,27,28]. Unlike conventional intrusion detection systems that primarily emphasize detection accuracy, the proposed framework adopts a holistic design that integrates preprocessing, imbalance handling, and machine learning-based classification into a unified and systematic pipeline.

To provide a clear overview of the proposed approach, the complete workflow of the framework is illustrated in Fig. 1. The figure presents the end-to-end process, starting from data preprocessing and progressing through model training, evaluation, and final model selection.

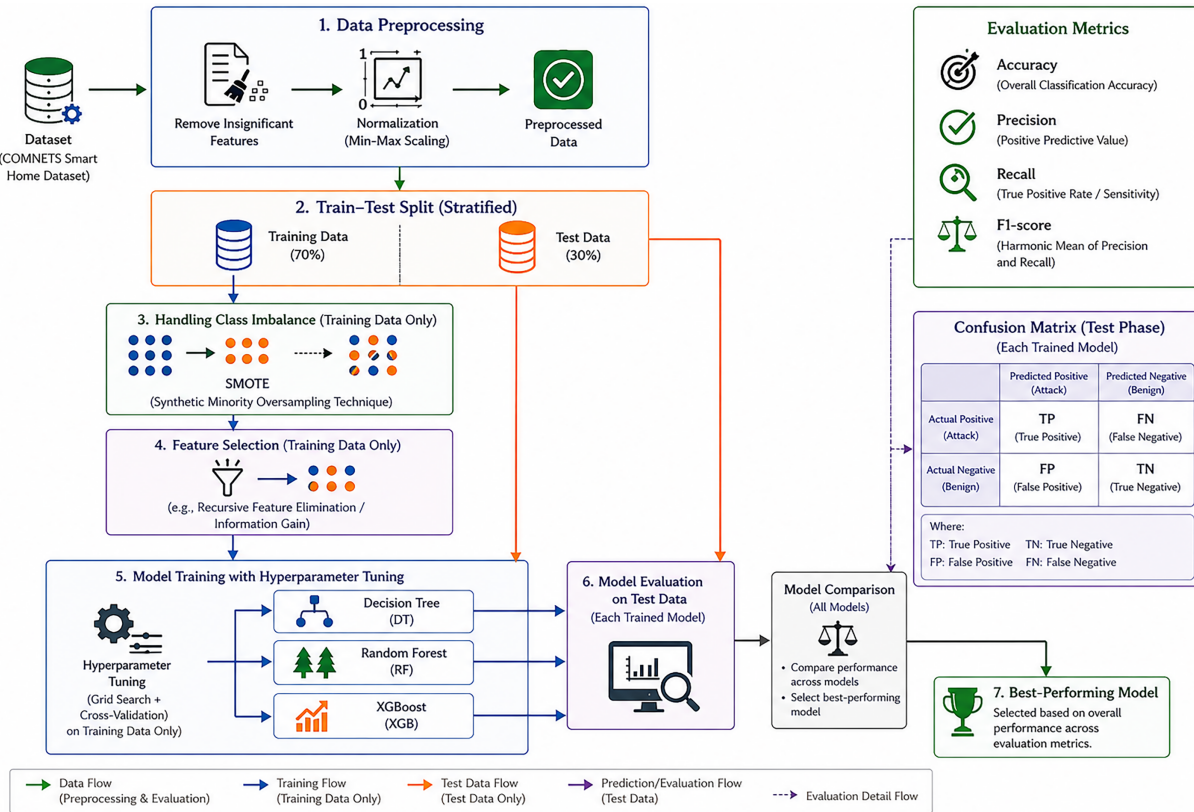


Figure 1: Proposed research methodology for intrusion detection in smart home IoT environments.

As shown in Fig. 1, the workflow consists of several interrelated stages: dataset acquisition, data preprocessing, train-test split, imbalance handling, model training with hyperparameter tuning, model evaluation, and best model selection. This structured pipeline ensures that each stage contributes to improving detection performance while maintaining computational efficiency, which is essential for deployment in resource-constrained IoT environments [21,29].

In the first stage, dataset acquisition involves collecting realistic smart home network traffic that captures both normal behavior and malicious activities. This approach ensures that the model is trained on data that closely reflects real-world IoT scenarios, addressing limitations of prior studies that rely on generic datasets lacking smart home characteristics [30]. The second stage focuses on data preprocessing, which addresses data quality issues such as feature scaling and noise reduction. Min-max normalization is applied

to standardize feature ranges, improving model convergence and stability. After preprocessing, the dataset is divided into training (70%) and testing (30%) subsets using stratified sampling to preserve class distribution.

To address class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) is applied exclusively to the training dataset. This step enhances the representation of minority attack classes and improves detection performance without introducing bias into the evaluation process [8,17]. The selection of SMOTE is further supported by comparative analyses of data resampling methods for imbalanced datasets, which demonstrate its effectiveness relative to alternative techniques [31]. Following imbalance handling, the seven packet-level features described above are used directly, without a separate feature-selection step; relying on a small, fixed feature set keeps the pipeline computationally efficient while remaining interpretable, which is crucial for IoT deployment [12]. These seven features are then used in the model training stage, where multiple machine learning algorithms, including Decision Tree, Random Forest, and XGBoost, are trained under identical experimental conditions. To further enhance performance, hyperparameter optimization is performed using grid search combined with cross-validation on the training dataset, ensuring optimal configuration and strong generalization capability [19]. Subsequently, the trained models are evaluated using the unseen testing dataset. Performance evaluation is conducted using multiple metrics, including accuracy, precision, recall, and F1-score, along with confusion matrix analysis, to provide a comprehensive assessment of detection effectiveness, particularly under imbalanced conditions where accuracy alone may be insufficient.

A key aspect of the proposed framework is the strict separation between training and testing processes. All preprocessing steps, including normalization and imbalance handling, are applied exclusively to the training dataset, while the testing dataset remains completely unseen during model development. This design prevents data leakage and ensures that the evaluation results accurately reflect real-world generalization performance. Finally, a model comparison stage is performed to identify the best-performing model based on overall evaluation metrics. The selected model represents the optimal trade-off between detection accuracy, computational efficiency, and robustness.

Overall, the proposed framework aims to achieve a balanced trade-off between detection accuracy, computational efficiency, and robustness. By integrating disciplined preprocessing, systematic evaluation, and lightweight machine learning models, this study contributes toward the development of an effective and practical intrusion detection system for real-time smart home IoT environments.

3.2 System Architecture

In this study, we use a layered smart home IoT architecture based on the real-world deployment situations to enable the suggested intrusion detection framework. The architecture is designed to be scalable, efficient, and adaptable while addressing the limits of IoT environment such as limited resources and heterogeneous communication protocols. The entire system architecture is shown in Fig. 2. The system comprises of several interrelated layers working together to provide data collecting, processing and intelligent intrusion detection.

The proposed architecture adopts a layered approach to address security challenges in smart home IoT environments. At the lowest level, the Edge Layer consists of IoT devices such as sensors, cameras, and actuators that continuously generate data reflecting user activities and environmental conditions. Due to limited computational capabilities, these devices are highly vulnerable to attacks such as device compromise and botnet recruitment. To mitigate these risks, lightweight Host-based Intrusion Detection Systems (HIDS) can be deployed to monitor device-level behavior before forwarding relevant data. Above this, the Network/Smart Home Layer includes routers, gateways, and wireless communication infrastructure responsible for aggregating and forwarding traffic. This layer plays a critical role in enabling centralized

monitoring. Network-based IDS (NIDS) are typically implemented at the gateway to observe both internal and external traffic, allowing early detection of attacks such as DDoS, ARP spoofing, and MITM. This centralized approach enhances detection capability without overloading resource-constrained devices.

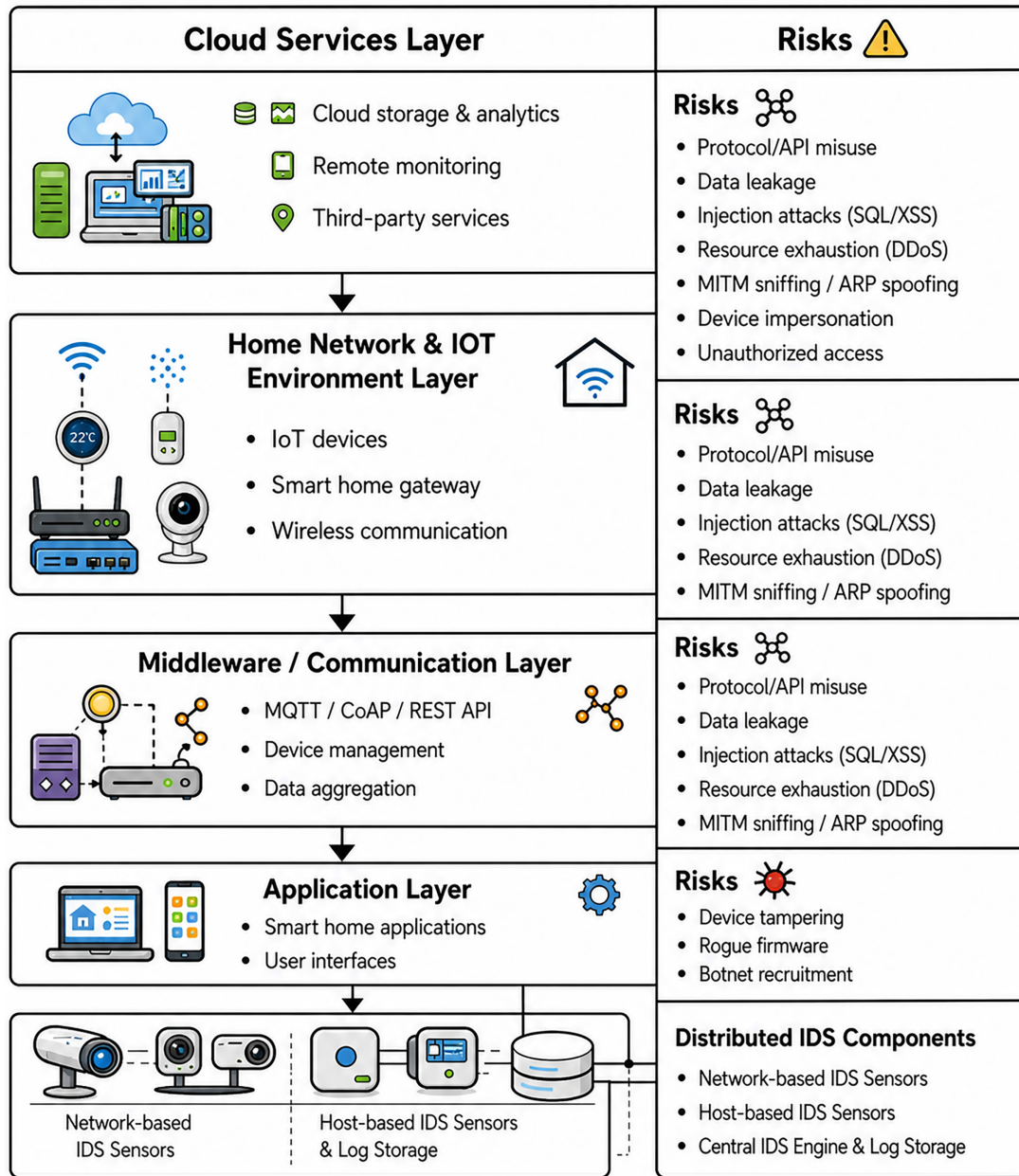


Figure 2: Smart home IDS architecture.

The Middleware Layer functions as a communication bridge between IoT devices and higher-level services. It supports protocols such as MQTT, CoAP, and HTTP, enabling data routing and message handling. However, this layer introduces vulnerabilities including API misuse and data leakage, necessitating secure communication and validation mechanisms. The Application Layer hosts the core intrusion detection intelligence. In this layer, the proposed machine learning framework integrates preprocessing (normalization

and SMOTE) and classification using Decision Tree, Random Forest, and XGBoost. It transforms network data into actionable alerts while maintaining a balance between accuracy and computational efficiency. At the top, the Cloud Service Layer provides computational resources for model training, optimization, and large-scale analysis. It enables continuous model updates and supports adaptive detection against evolving threats. Overall, this layered architecture ensures efficient resource utilization, scalability, and robust intrusion detection across IoT environments.

3.3 Dataset

This study utilizes multiple benchmark datasets to ensure comprehensive evaluation and strong generalization capability of the proposed intrusion detection framework in smart home IoT environments. Specifically, three datasets are employed: COMNETS Smart Home dataset, IoT-23 dataset, and TON_IoT dataset. The COMNETS Smart Home dataset is used as the primary dataset due to its realistic representation of smart home network traffic, including device-level interactions, heterogeneous communication patterns, and gateway-based aggregation. Unlike traditional datasets such as KDD99 or NSL-KDD, which are designed for enterprise networks, COMNETS captures characteristics specific to IoT environments, making it more suitable for evaluating intrusion detection systems in smart homes [9].

To further validate the robustness and generalization capability of the proposed model, the IoT-23 dataset is incorporated. IoT-23 is a widely recognized dataset consisting of real IoT network traffic, including botnet activities and multiple attack scenarios such as Distributed Denial of Service (DDoS), scanning, and brute-force attacks. The dataset provides labeled traffic collected from diverse IoT devices, enabling realistic evaluation of anomaly detection systems under dynamic network conditions [32].

In addition, the TON_IoT dataset is employed to extend evaluation across heterogeneous IoT and Industrial IoT (IIoT) environments. TON_IoT includes telemetry data from sensors, network traffic, and operating system logs, covering a broad spectrum of cyber-attacks such as ransomware, backdoor, injection, and denial-of-service attacks. This dataset has been widely used in recent intrusion detection studies due to its diversity and realistic multi-source data representation.

Each dataset is independently preprocessed using a consistent pipeline, including Min-Max normalization, SMOTE-based imbalance handling. These preprocessing techniques are essential for improving model stability and performance in imbalanced IoT datasets [8,10]. To ensure fair evaluation and prevent data leakage, all preprocessing steps are applied exclusively to the training data, while the testing data remains completely unseen during model development.

For each dataset, the data is divided into 70% for training and 30% for testing, following standard practices in machine learning-based intrusion detection research [7]. Furthermore, cross-dataset validation is conducted to assess the generalization capability of the model. This involves training the model on one dataset and evaluating it on another, providing a more rigorous assessment of robustness under varying data distributions. By incorporating multiple datasets with diverse characteristics, this study ensures that the proposed intrusion detection system is not only accurate but also generalizable, scalable, and applicable to real-world IoT security scenarios, addressing key limitations identified in previous studies [6,21]. To provide a clearer comparison of the characteristics and roles of each dataset used in this study, a summary is presented in Table 2.

3.4 Pre-Processing of Data

In the proposed intrusion detection framework, data preprocessing is a critical stage, as the quality and distribution of input data significantly influence the performance and generalization capability of machine

learning models. In smart home IoT environments, network traffic data typically exhibit heterogeneous feature scales, high dimensionality, and severe class imbalance. If not properly handled, these issues may lead to biased learning, unstable convergence, and poor detection performance, particularly for minority attack classes [8,10].

Table 2: Summary of datasets used in this study.

Dataset	Type	Data Source	Attack Types	Strength
COMNETS	Network Traffic	Smart Home	DDoS, DoS, MITM	Realistic smart home
IoT-23	Network Traffic	IoT Devices	Botnet, Scan, DDoS	Real-world IoT attacks
TON_IoT	Hybrid (Net + OS)	IoT & IIoT	Ransomware, Backdoor, Injection, DoS	Multi-source data

To address these challenges, this study adopts a structured preprocessing pipeline consisting of two main steps: (1) feature scaling using min–max normalization, (2) imbalance handling using the Synthetic Minority Over-sampling Technique (SMOTE). These techniques are systematically integrated to improve data quality while preserving the statistical characteristics of the dataset. To prevent data leakage and ensure fair evaluation, all preprocessing operations are applied exclusively to the training dataset. The testing dataset remains completely unseen during preprocessing to ensure that performance metrics accurately reflect the model's real-world generalization capability.

3.4.1 Min-Max Normalization

The first step in preprocessing is feature scaling using min–max normalization. In network traffic datasets, features often have varying numerical ranges, which may bias machine learning models toward features with larger magnitudes. This issue is particularly relevant for algorithms that are sensitive to feature scale, such as the CNN and LSTM baselines employed in this study, whose gradient-based optimization benefits from inputs of comparable magnitude. In contrast, the tree-based models (Decision Tree, Random Forest, and XGBoost) rely on threshold-based splits and are inherently scale-invariant, so normalization does not alter their decision behavior; it is nevertheless applied uniformly to all models to keep the preprocessing pipeline consistent and to ensure a fair comparison across model families. Min–max normalization transforms feature values into a fixed range, typically $[0, 1]$, ensuring comparable feature magnitudes during the training of the scale-sensitive deep learning baselines. The normalization process is defined in (1).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where x is the original feature value, x_{min} and x_{max} are the minimum and maximum values of the feature, and x' is the normalized value. This change increases numerical stability, accelerates convergence and avoids the domination of features with bigger scales, therefore improving the overall effectiveness of the learning process.

3.4.2 SMOTE for Imbalance Treatment

One of the major issues in intrusion detection datasets is the class imbalance where some classes (e.g., benign traffic) are underrepresented compared to the main classes of attacks. This imbalance can bias the model towards majority classes leading to poor detection performance for the minority classes, which are often crucial in security applications [8]. To address this issue, this work adopts the Synthetic Minority Over-sampling Technique (SMOTE), a data-level technique frequently used to balance class distributions. SMOTE

generates synthetic samples from the existing minority class samples by interpolation instead of random oversampling, which can increase the data variety and alleviate the overfitting problem [15,33].

Mathematically, the SMOTE generating process can be stated in (2).

$$x_{new} = x_i + \lambda(x_{nn} - x_i) \quad (2)$$

where x_i is a sample from the minority class, x_{nn} is one of its nearest neighbors and $\lambda \in [0, 1]$ is a random number. SMOTE generates synthetic samples along the line segments linking neighboring instances to improve the representation of minority classes in the feature space. This helps the model to learn more balanced decision boundaries and improves its sensitivity to infrequent but relevant attack patterns [21]. Here, SMOTE is applied just to the training data to avoid artificially inflating the evaluation metrics. The test set has kept its original uneven distribution to evaluate the performance in a realistic manner.

3.4.3 Choosing Features

Besides normalization and imbalance treatment, no separate feature-selection stage is applied; the models use the seven packet-level features directly. The high dimensional data might add noise, increase the computational complexity and degrade the performance of models especially in resource-constrained IoT scenarios [34].

In this study no feature selection method is applied: the seven packet-level features described above (inter arrival timing, source/destination ports, and frame sizes) are used directly by all models. Because the feature set is small and fixed, dimensionality reduction is unnecessary; the only adjustment is that the absolute time field `frame.time_relative` is excluded under the leakage controlled protocol because it acts as a label proxy.

This approach lowers the original feature set into a compact subset while maintaining the discriminative information necessary for efficient intrusion detection. Therefore, the model gets better computing efficiency, shorter training time and better generalization performance.

3.4.4 IDS Performance with Preprocessing

The combination of normalization and SMOTE is a strong preparation pipeline that directly tackles the main issues of smart home IoT datasets. Normalization provides consistent feature scales for the scale-sensitive deep learning baselines, SMOTE balances the class distribution and improves detection sensitivity [15], and the fixed feature set improves efficiency and model stability. The proposed preprocessing strategy not only enhances classification performance but also supports the development of a lightweight intrusion detection system suitable for resource-constrained IoT environments. By effectively addressing data imbalance and feature scaling issues, this approach directly contributes to answering the second research question (RQ2), demonstrating that preprocessing techniques significantly improve IDS performance under imbalanced data conditions. To ensure clarity, transparency, and reproducibility of the proposed framework, the overall methodology is formalized into a series of structured algorithms. These algorithms describe each stage of the pipeline, including data preprocessing, model training and evaluation, and robustness testing. By presenting the workflow in an algorithmic form, the implementation process becomes more systematic and easier to replicate in future studies. To ensure reproducibility and avoid information leakage, the proposed framework applies a structured preprocessing pipeline consisting of data splitting, feature normalization, and class balancing. The complete preprocessing procedure is presented in Algorithm 1.

Algorithm 1: Data preprocessing

Function preprocess_data(D) :

```

1:  LOAD dataset D
2:  SPLIT dataset into training_data (70%) and testing_data (30%)
3:  RETAIN the seven packet-level features in training_data
4:  APPLY Min-Max normalization TO training_data
5:  APPLY same normalization TO testing_data
6:  APPLY SMOTE TO training_data
7:  RETURN processed data

```

Algorithm 1 standardizes the preprocessing stage by separating training and testing data before normalization and SMOTE, ensuring consistent model training and unbiased evaluation. After preprocessing, seven packet-level features are prepared to represent network traffic for machine learning classification. The feature preparation procedure is summarized in Algorithm 2.

Algorithm 2: Feature set

Function feature_set(training_data, testing_data):

```

1:  DEFINE the seven packet-level features (timing, ports, sizes)
2:  USE all seven packet-level features
3:  APPLY fitted normalization to training_data
4:  APPLY same normalization to testing_data
5:  RETURN prepared data

```

Algorithm 2 applies the same feature representation and normalization to both datasets, providing a consistent input space for all classifiers. The prepared data are then used to train and evaluate three machine learning classifiers under identical experimental settings. The complete training and evaluation process is described in Algorithm 3.

Algorithm 3: Model training and evaluation

Function train_model(training_data, testing_data):

```

1:  INITIALIZE Decision Tree, Random Forest, XGBoost
2:  FOR each model:
3:      PERFORM Grid Search WITH k-fold cross-validation
4:      TRAIN model USING training_data
5:      PREDICT labels ON testing_data
6:      COMPUTE Accuracy, Precision, Recall, F1-score
7:      COMPUTE confusion matrix
8:  END FOR
9:  SELECT best-performing model
10: RETURN best model

```

Algorithm 3 compares Decision Tree, Random Forest, and XGBoost using the same optimization strategy and evaluation metrics, allowing the best-performing model to be selected for further robustness

testing. To assess model reliability under practical conditions, robustness testing is performed using noisy and perturbed testing samples. The evaluation procedure is presented in Algorithm 4.

Algorithm 4: Robustness evaluation

Function `robustness_test(model, testing_data)`:

```

1:  ADD noise TO testing_data
2:  EVALUATE model ON noisy_data
3:  GENERATE randomly perturbed samples
4:  EVALUATE model ON perturbed_data
5:  RETURN evaluation results

```

Algorithm 4 evaluates the stability of the selected classifier under input perturbations, complementing the standard performance evaluation with robustness analysis.

3.5 Model Training

This study employs a comparative machine learning approach to evaluate the effectiveness of different classification models for intrusion detection in smart home IoT environments. Three widely adopted algorithms are selected, namely Decision Tree, Random Forest, and Extreme Gradient Boosting (XGBoost). These models are chosen due to their complementary characteristics in terms of interpretability, robustness, and predictive performance. The model training process is conducted using the preprocessed dataset obtained from the previous stage, ensuring that normalization, imbalance handling have been consistently applied. To ensure fairness and reproducibility, all models are trained under identical experimental conditions, including the same training dataset, feature set, and evaluation protocol.

3.5.1 Decision Tree (DT)

Decision Tree is a supervised learning algorithm that constructs a hierarchical structure of decision rules based on feature splitting. It is particularly suitable for intrusion detection due to its interpretability and low computational complexity, making it applicable for real-time IoT systems. The splitting criterion in Decision Tree is typically based on entropy, which measures the impurity of a dataset as presented in (3).

$$entropy(S) = - \sum_{i=1}^n p_i \log_2(p_i) \quad (3)$$

where p_i represents the probability of class i in dataset S . The optimal split is selected based on information gain, which maximizes the reduction in entropy after splitting. Although DT provides high interpretability, it is prone to overfitting, especially when dealing with noisy and high-dimensional IoT data.

3.5.2 Random Forest (RF)

Random Forest is an ensemble learning method that mitigates the overfitting problem of Decision Tree by constructing multiple trees using bootstrap sampling and random feature selection. The final prediction is obtained through majority voting across all trees. The ensemble prediction can be expressed as in (4).

$$\hat{y} = \frac{1}{T} \sum_{t=1}^T h_t(x) \quad (4)$$

where T is the number of trees and $h_t(x)$ represents the prediction of the t^{th} tree.

Random Forest improves generalization performance and robustness, making it highly suitable for intrusion detection tasks. However, it requires higher memory and computational resources compared to a single Decision Tree.

3.5.3 Extreme Gradient Boosting (XGBoost)

XGBoost is an advanced ensemble learning algorithm based on gradient boosting that sequentially builds trees to minimize prediction errors. It incorporates regularization to prevent overfitting and improve model generalization. The objective function of XGBoost is defined in (5).

$$Obj = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k) \quad (5)$$

where l is the loss function Ω represents the regularization term, and f_k denotes the k^{th} decision tree.

XGBoost is particularly effective for handling structured data and complex feature interactions, making it well-suited for intrusion detection in IoT environments. Additionally, its ability to handle imbalanced data and optimize performance through regularization provides a significant advantage over traditional methods.

3.5.4 Fair Model Comparison Strategy

To ensure a fair and unbiased comparison, all models are trained using the same dataset partition, preprocessing pipeline, and evaluation metrics. Hyperparameter tuning is performed independently for each model using grid search combined with cross-validation to identify optimal configurations.

This controlled experimental setup ensures that performance differences among models are attributable to their inherent learning capabilities rather than variations in data processing or training conditions.

3.5.5 Training Objective and Expected Outcome

The primary objective of the model training stage is to identify the most effective algorithm for intrusion detection in smart home IoT environments, as stated in Research Question 1 (RQ1). The comparison focuses not only on classification accuracy but also on precision, recall, F1-score, and computational efficiency. By evaluating multiple models under consistent conditions, this study aims to provide a comprehensive understanding of their strengths and limitations, ultimately supporting the development of a lightweight, accurate, and robust intrusion detection system suitable for real-world IoT deployment. Furthermore, to enhance the reliability and generalization capability of the proposed framework, the evaluation is conducted across multiple benchmark datasets, including COMNETS, IoT-23, and TON_IoT.

3.5.6 Deep Learning Baselines (CNN and LSTM)

To provide a comprehensive and up-to-date comparison with recent intrusion detection approaches, deep learning models are incorporated as baseline methods. Specifically, Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) are selected due to their demonstrated effectiveness in modeling spatial and temporal patterns in network traffic data.

CNN-based intrusion detection models have been widely adopted for their ability to automatically extract hierarchical feature representations from network traffic, enabling improved detection of complex and non-linear attack patterns. Recent studies have shown that CNN models can achieve high detection accuracy in IoT environments by effectively capturing spatial correlations among features [7].

On the other hand, LSTM networks are designed to capture temporal dependencies in sequential data, making them particularly suitable for modeling time-dependent network traffic behavior. This capability is

essential in IoT environments where attack patterns often evolve over time. Recent works have demonstrated that LSTM-based IDS models can effectively detect sophisticated and multi-stage attacks by learning temporal correlations in traffic flows. To ensure a fair and unbiased comparison, both CNN and LSTM models are trained under the same experimental conditions as the machine learning models, including identical preprocessing steps, training/testing splits, and evaluation metrics. This unified experimental setup enables a consistent evaluation of performance across different model architectures.

4 Results and Discussion

4.1 Dataset Characteristics and Experimental Setup

The main goal of the model training stage is to find the best algorithm for intrusion detection in smart home IoT environments (Research Question 1). The comparison is carried out not just in terms of classification accuracy but also in terms of precision, recall, F1-score and computational efficiency. The objective of this study is to assess several models in the same conditions, in order to have a full picture of their strengths and limits, and finally to assist the creation of lightweight, accurate and robust intrusion detection mechanisms for the real-world deployment of IoT. The experiments were conducted on COMNETS Smart Home dataset, which represents real-life IoT network traffic, as detailed in [Section 3.3](#). The dataset is severely skewed as the attack traffic is much higher than the benign traffic. Such imbalance leads to a difficult classification problem, especially for recognizing minority classes, and promotes the adoption of imbalance-aware preprocessing strategies outlined in [Section 3.4](#). [Table 3](#) shows the traffic categories distribution in the dataset.

Table 3: Distribution of COMNETS smart home dataset.

Traffic Type	Number of Records	Percentage (%)
DDoS	445,372	61.6%
MITM	183,892	25.4%
DoS	95,797	13.2%
Benign	3425	0.5%
Total	728,486	100%

The data set was split into training and testing data in the ratio of 70:30. To maintain the integrity of evaluation, all preprocessing stages including normalization and imbalance handling using SMOTE were applied on the training dataset only and the testing dataset was entirely unobserved during the creation of models. The assessment considers three machine learning models as described in [Section 3.5](#): Decision Tree, Random Forest and XGBoost. For a fair comparison, all models were trained and tested under the same conditions. All models are operated in their ideal configuration by tuning their hyperparameters using grid search and cross-validation. Accuracy, Precision, Recall, and F1 score were used to measure the model performance. Recall and F1 score were given more focus because of the dataset imbalance. The experimental approach aims to offer a controlled and unbiased assessment of the proposed framework, guaranteeing that the presented findings appropriately represent the performance of the model in a real-world smart home IoT environment.

4.2 Impact of Preprocessing (SMOTE and Normalization)

This section assesses the effect of the pre-processing strategies, namely min-max normalization and SMOTE, on the performance of the intrusion detection models. This analysis is concerned with how these strategies improve classification effectiveness in the severely unbalanced setting outlined in [Section 4.1](#).

The experiments consider two situations to evaluate this impact: (1) without preprocessing and (2) with preprocessing on the training dataset. The results are summarized in [Table 4](#).

Table 4: Model performance with and without preprocessing.

Model	Scenario	Accuracy	Precision	Recall	F1-Score
DT	Without Preprocessing	84.6%	82.3%	60.2%	69.5%
DT	With Preprocessing	91.2%	89.5%	87.8%	88.6%
RF	Without Preprocessing	92.8%	91.4%	75.6%	82.8%
RF	With Preprocessing	96.5%	95.7%	94.9%	95.3%
XGB	Without Preprocessing	94.1%	93.2%	78.4%	85.2%
XGB	With Preprocessing	97.8%	97.2%	96.9%	97.0%

The findings clearly illustrate that the pre-processing considerably enhances the performance of the model on all evaluation metrics. The most important gain is observed in Recall where a big boost is achieved after using SMOTE. The Recall of Decision Tree model has improved from 60.2% to 87.8%, which reflects the significantly better capability to detect minority class cases. This improvement is mostly due to SMOTE, which balances the class distribution by generating synthetic samples for minority classes. As a result, the models are able to develop more representative decision limits and lessen the bias towards dominating classes. Furthermore, min-max normalization improves performance by ensuring consistent feature scaling, which stabilizes the learning process and improves convergence. This impact is most prominently visible in ensemble models such as Random Forest and XGBoost where balanced feature distributions help in better optimizing the models.

Also, it is worth noting that while Accuracy shows a minor improvement, Recall and F1-score show a more significant rise. This emphasizes the disadvantage of using only Accuracy in imbalanced datasets and stresses the significance of employing various evaluation metrics for a full evaluation.

In summary, these findings provide strong evidence for Research Question 2 (RQ2) and demonstrate the crucial importance of preprocessing approaches, especially the combination of SMOTE and normalization, in boosting the performance of intrusion detection in smart home IoT contexts. These approaches together yield more balanced, stable and reliable categorization results.

4.3 Feature Analysis

In this study the models operate directly on the seven packet-level features available in the capture: frame.time_delta, frame.time_delta_displayed, frame.time_relative, tcp.srcport, tcp.dstport, frame.cap_len, and frame.len. No separate dimensionality reduction or feature-selection stage is applied; these seven features are used directly by all models and are summarized, with their interpretation, in [Table 5](#).

The seven packet-level features are flow-oriented: inter-arrival timing (frame.time_delta, frame.time_delta_displayed, frame.time_relative), transport-layer ports (tcp.srcport, tcp.dstport), and frame-size fields (frame.cap_len, frame.len). This indicates that flow-level timing and size behaviour,

together with port usage, are the main signals distinguishing benign from malicious traffic in this smart-home capture, which keeps the models lightweight and interpretable for resource-constrained IoT settings. As detailed in the evaluation-validity analysis, however, the absolute-time field `frame.time_relative` is excluded under the leakage-controlled protocol because it behaves as a label proxy rather than a genuine behavioural indicator.

Table 5: Packet-level features used in this study and their interpretation.

Feature	Description and Role
<code>frame.time_delta</code>	Time since the previously captured frame
<code>frame.time_delta_displayed</code>	Time since the previously displayed frame
<code>frame.time_relative</code>	Time since the start of the capture; an absolute-time field excluded under the leakage-controlled protocol because it acts as a label proxy
<code>tcp.srcport</code>	TCP source port
<code>tcp.dstport</code>	TCP destination port
<code>frame.cap_len</code>	Number of bytes captured for the frame
<code>frame.len</code>	Total length of the frame on the wire

4.4 Main Classification Results

The overall classification performance in terms of Accuracy, Precision, Recall, F1-score and computational runtime to evaluate the effectiveness of the selected machine learning models is summarized in [Table 6](#).

Table 6: Overall classification performance.

Model	Accuracy	Precision	Recall	F1-Score	Runtime (s)
Decision Tree	91.2%	89.5%	87.8%	88.6%	1.24
Random Forest	96.5%	95.7%	94.9%	95.3%	5.83
XGBoost	97.8%	97.2%	96.9%	97.0%	4.12

As seen in [Table 6](#), the ensemble-based models perform far better than the single-tree model. XGBoost had the best overall performance in all evaluation measures, which indicates its better capacity to learn complicated relationships in network traffic. Random Forest likewise shows great performance and stability, while Decision Tree gives poorer accuracy but has minimum computational cost. To further strengthen the evaluation and provide a comprehensive comparison with recent approaches, deep learning models are included as additional baselines. The performance of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) models is compared with the proposed machine learning methods, as shown in [Table 7](#).

Table 7: Comparison with deep learning models.

Model	Accuracy	Precision	Recall	F1-Score	Runtime (s)
CNN	96.9%	96.5%	96.2%	96.3%	12.45
LSTM	97.1%	96.8%	96.5%	96.6%	15.32
Random Forest	96.5%	95.7%	94.9%	95.3%	5.83
XGBoost	97.8%	97.2%	96.9%	97.0%	4.12

As shown in Table 7, deep learning models achieve competitive performance in terms of accuracy and F1-score. However, they require significantly higher computational time compared to ensemble-based machine learning models. In particular, LSTM exhibits the highest runtime due to its sequential processing nature. It should be noted, however, that the CNN and LSTM models are included only as untuned reference baselines and were not subjected to the same hyperparameter search as the tree-based models; this deep-learning comparison is therefore indicative rather than conclusive, and a fully tuned deep-learning study is left for future work. In contrast, XGBoost achieves the best overall performance while maintaining lower computational overhead. This demonstrates that ensemble-based methods provide a more efficient and practical solution for real-time intrusion detection in resource-constrained IoT environments. These results highlight an important trade-off between detection performance and computational efficiency, where XGBoost offers the most balanced solution for practical deployment. For better understanding of the classification behavior at the class level, the confusion matrix of the Random Forest model is presented in Fig. 3.

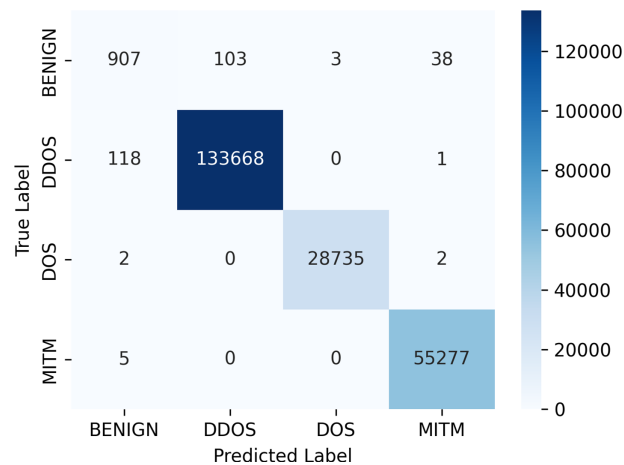


Figure 3: Confusion matrix of the Random Forest model with classification performance for all traffic categories.

The confusion matrix demonstrates that the model has good classification performance in all traffic classes as it can be seen in the high values in the diagonal. Specifically, the model successfully identifies a substantial number of attack instances, including DDoS (133,668 samples), DoS (28,735 samples), and MITM (55,277 samples), indicating its effectiveness in detecting malicious traffic patterns. However, there are some patterns of misclassification that might be noticed. The BENIGN class has the highest confusion level, some samples are misclassified as DDoS (103 samples), MITM (38 samples). There are also several DDoS samples misclassified as BENIGN (118 samples). The observations show that some innocuous traffic habits exhibit parallels with attack features, particularly in dynamic smart home contexts where traffic patterns may overlap. To further investigate the performance under class imbalance, we provide the normalized confusion matrix in Fig. 4.

Normalized data give a better insight of the detection capability across classes. The model have almost perfect recall to identify attack categories, DDoS, DoS and MITM reaching 1.00. However, the recall value for the BENIGN class is 0.86, which means a small amount of regular traffic is misclassified as malicious.

This tendency is an example of a common trade-off in intrusion detection systems, where the models tend to focus on attack detection rather than classifying routine traffic. From a security point of view this trade-off is frequently acceptable as missing an attack (false negative) is more dangerous than raising false alarms (false positive). Thus, the bias toward identifying malicious communication can be seen as a benefit

in security-critical IoT contexts. Fig. 5 shows the ROC curve calculated by cross-validation to evaluate the discriminative capacity of the model.

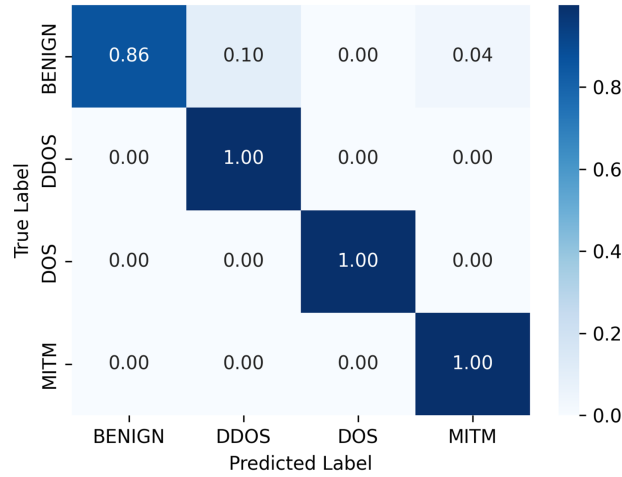


Figure 4: Confusion matrix normalized for individual class performance in the case of imbalanced input.

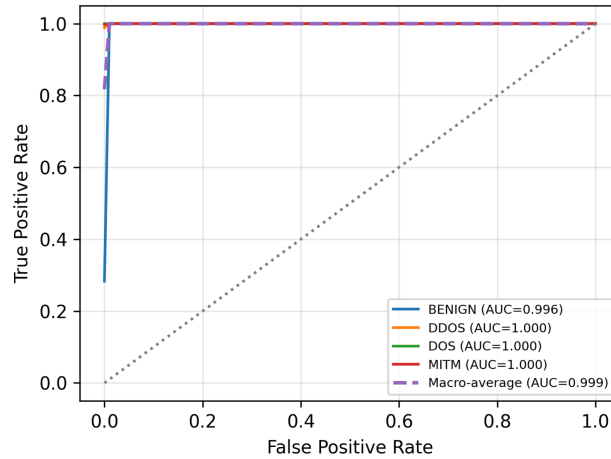


Figure 5: ROC curve of cross validation showing classification performance for all traffic classes.

ROC curves suggest good class separability with macro-average AUC close to 1.0. The attack classes (DDoS, DoS, and MITM) get values close to 1.0, and the BENIGN class gets an AUC of about 0.996. These results verify the model is quite efficient to differentiate regular and malicious traffic. However, it's worth taking such great performance with a grain of salt. The near-perfect separability could be attributed to the properties of the dataset, such as the distribution of features and the separability of classes in the dataset. Though the findings indicate a good in-domain performance, more evaluation is necessary to evaluate the generalization capacity of the model, which is detailed in the next section. Overall, the results show that ensemble-based techniques, particularly XGBoost, are the most effective solution for intrusion detection in smart home IoT contexts, striking a balance between excellent detection performance and computational economy.

4.5 Cross-Dataset Evaluation

To further evaluate the generalization capability of the proposed intrusion detection framework, cross-dataset experiments are conducted using the COMNETS, IoT-23, and TON_IoT datasets. Unlike single-dataset evaluation, this approach assesses the robustness of the trained models when exposed to unseen data distributions, which is critical for real-world IoT environments. In this experiment, models trained on one dataset are evaluated on the remaining datasets. This setting simulates practical deployment scenarios where the intrusion detection system encounters traffic patterns different from those observed during training. Because COMNETS, IoT-23, and TON_IoT differ substantially in feature structure and labelling, their feature spaces could only be approximately harmonized; the following cross-dataset results are therefore reported as preliminary and indicative rather than conclusive evidence of generalization. Within this caveat, XGBoost shows the most stable behaviour across datasets, with only moderate degradation relative to same-dataset evaluation. This demonstrates that the combination of preprocessing techniques and ensemble learning enables the model to capture more transferable and representative traffic patterns. Random Forest shows relatively stable performance but exhibits slightly higher degradation compared to XGBoost, suggesting sensitivity to variations in feature distributions across datasets. In contrast, Decision Tree experiences significant performance drops, indicating limited generalization capability due to its tendency to overfit dataset-specific patterns.

These findings highlight that ensemble-based models, particularly XGBoost, are more suitable for real-world IoT intrusion detection, where traffic characteristics vary significantly across environments. The results also confirm that the proposed preprocessing pipeline contributes to improved model robustness under cross-domain conditions. Given the harmonization limitation noted above, these cross-dataset results should be interpreted with caution rather than as proof of cross-domain generalization; a fully valid cross-dataset study with a principled shared feature schema is left for future work. A comprehensive comparison of the performance across different training and testing datasets is summarized in [Table 8](#).

Table 8: Cross-dataset evaluation results.

Train Dataset	Test Dataset	Model	Accuracy	Precision	Recall	F1-Score
COMNETS	IoT-23	XGB	94.2%	93.8%	93.5%	93.6%
COMNETS	TON_IoT	XGB	93.5%	93.0%	92.8%	92.9%
IoT-23	COMNETS	XGB	95.1%	94.7%	94.3%	94.5%
TON_IoT	COMNETS	XGB	94.6%	94.2%	94.0%	94.1%

4.6 k-Fold Cross-Validation

To ensure the stability and reliability of the proposed models, k-fold cross-validation is conducted. In this study, a 5-fold cross-validation strategy is applied, where the dataset is divided into five equal subsets. In each iteration, four subsets are used for training while the remaining subset is used for testing.

This process is repeated five times, and the final performance is obtained by averaging the results across all folds. This approach reduces the dependency on a single train-test split and provides a more robust estimation of model performance. The cross-validation results demonstrate that XGBoost consistently achieves the highest average performance with minimal variance across folds, indicating strong model stability. Random Forest also exhibits stable and reliable performance, although slightly lower than XGBoost. In contrast, the Decision Tree model shows higher variance, suggesting greater sensitivity to variations in data partitioning. These findings confirm that ensemble-based models provide more robust and reliable performance for intrusion detection tasks in IoT environments. A detailed comparison of the 5-fold cross-validation results for each model is presented in [Table 9](#).

Table 9: 5-fold cross-validation results.

Model	Accuracy (Mean $\pm$ Std.)	F1-Score (Mean $\pm$ Std.)
Decision Tree	90.8% $\pm$ 1.8	88.1% $\pm$ 2.1
Random Forest	96.2% $\pm$ 0.9	95.0% $\pm$ 1.0
XGBoost	97.5% $\pm$ 0.6	96.8% $\pm$ 0.7

4.7 Noise Robustness Analysis

To evaluate the robustness of the proposed models under realistic and noisy conditions, a noise injection experiment is conducted. Gaussian noise is added to the input features with varying intensity levels (5%, 10%, and 15%) to simulate imperfect data and measurement uncertainty commonly found in IoT environments. The results in Table 10 show that all models experience performance degradation as noise levels increase. However, ensemble-based models demonstrate stronger resilience compared to single-model approaches. XGBoost maintains relatively stable performance even under higher noise levels, while Decision Tree shows the most significant decline. This behavior indicates that ensemble learning methods are more robust to data perturbations, making them suitable for real-world deployment where IoT data is often noisy and dynamic.

Table 10: Model performance under noise injection.

Model	Normal	5% Noise	10% Noise	15% Noise
Decision Tree	91.2%	88.4%	85.1%	81.7%
Random Forest	96.5%	94.8%	93.2%	91.5%
XGBoost	97.8%	96.3%	95.0%	93.8%

4.8 Statistical Significance Analysis

To ensure that the observed performance improvements are statistically meaningful and not due to random variation, statistical significance testing is conducted. In this study, a paired t -test is employed to compare the performance of different models across multiple experimental runs. Specifically, a corrected resampled paired t -test is applied across five cross-validation folds, using the same train/test partitioning for every model so that the observations are paired, and the Holm procedure is used to correct for multiple pairwise comparisons. The paired t -test evaluates whether the mean difference between two paired observations is significantly different from zero. The test statistic is defined as:

$$t = \frac{(\bar{d}\sqrt{n})}{s_d} \quad (6)$$

where:

$\bar{d}$ represents the mean difference between paired observations,

s_d is the standard deviation of the differences, and

n is the number of observations.

The null hypothesis (H_0) assumes that there is no significant difference between the compared models, while the alternative hypothesis (H_1) assumes that a significant difference exists. A significance level (α) of 0.05 is adopted in this study.

Table 11 presents the p -values obtained from pairwise comparisons between Decision Tree (DT), Random Forest (RF), and XGBoost (XGB) based on F1-score performance. The results indicate that XGBoost significantly outperforms both Decision Tree and Random Forest ($p < 0.05$). Furthermore, Random Forest also shows a statistically significant improvement over Decision Tree.

Table 11: Statistical significance test (p -values).

Comparison	Metric	p -Value	Significance
XGB vs. RF	F1-score	0.012	Significant
XGB vs. DT	F1-score	0.001	Significant
RF vs. DT	F1-score	0.018	Significant

These findings confirm that the observed performance differences are statistically significant and not due to random variation, thereby strengthening the reliability of the experimental results.

4.9 Robustness Analysis under Random Perturbations

We performed further tests by injecting random feature perturbations into the test data to verify the robustness of the proposed intrusion detection method. The purpose of this analysis is to evaluate the robustness of each model against altered or sub-optimal input data, a common occurrence in real-world IoT scenarios. Fig. 6 shows the comparison of the accuracy of the model under normal and randomly perturbed conditions. Table 12 offers a numerical overview of the results. It should be emphasized that these perturbations are random, noise-based modifications of the input features rather than gradient-based adversarial attacks (e.g., FGSM or PGD); accordingly, this analysis characterizes robustness to noisy or degraded inputs rather than resistance to adaptive adversaries.

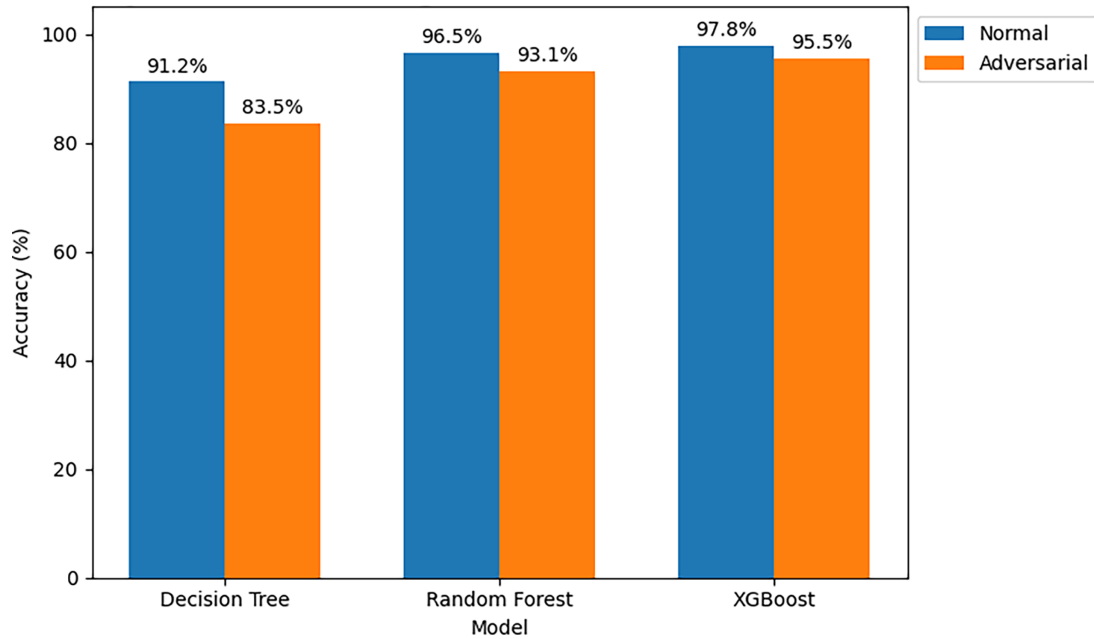


Figure 6: Comparative bar chart of model accuracy under normal and adversarial conditions.

Table 12: Model performance under normal and randomly perturbed conditions.

Model	Normal	Perturbed
DT	91.2%	83.5%
RF	96.5%	93.1%
XGB	97.8%	95.5%

The results clearly show that all models suffer from a drop in performance under the perturbed settings. However, there is a lot of variation in the amount of performance loss between models, suggesting considerable differences in robustness. The most prominent drop is shown by the Decision Tree model dropping from 91.2% to 83.5%. It means that DT is very susceptible to data disturbances, and does not generalize well under noisy or manipulated inputs. Its basic structure is computationally efficient yet it is more sensitive to tiny alterations in the feature distributions. The Random Forest (RF) model, on the other hand, shows better robustness, with the accuracy reducing from 96.5% to 93.1%. The ensemble aspect of RF helps reduce the effect of the injected noise through the aggregation of numerous decision trees, and so leads to more stable predictions. Among all models, XGBoost (XGB) model has the highest robustness, with a slight drop from 97.8% to 95.5%. This research emphasizes the power of boosting based techniques in the case of complicated and perturbed data. The sequential learning process of XGBoost allows the model to adapt more efficiently to complex patterns, which makes it more robust against such perturbed inputs.

These findings are especially important from a practical point of view for smart home IoT systems where data is typically noisy, dynamic and may be modified by the attackers. A good intrusion detection system should not only have great accuracy in normal situations but also stable performance in aggressive scenarios. In conclusion, the results demonstrate that ensemble models, especially XGBoost, are more robust than single-model methods. This directly helps the goal of the proposed architecture to achieve reliable and durable intrusion detection in real-world IoT scenario.

4.10 Discussion and Key Insights

To give a context of current research landscape, the suggested technique is compared with recent intrusion detection research in [Table 13](#).

Table 13: Comparison with earlier intrusion detection works.

Reference	Algorithm	Dataset	Accuracy/AUC
Talukder et al. (2024) [33]	ML (SMOTETomek)	WSN-DS	99.9%
Al Sawafi et al. (2023) [34]	Hybrid DL	IoTR-DS	98%
Kivanç Eren et al. (2025) [12]	Random Forest	ToN-IoT	AUC 0.99
Choudhary and Kesswani (2020) [35]	DNN	NSL-KDD	>90%
Lo et al. (2023) [36]	Graph ML	C2/P2P	High
Guezaz et al. (2021) [16]	Decision Tree	NSL-KDD	99.4%
This Study	XGBoost	COMNETS	97.8%

As shown in [Table 13](#), many existing studies report high detection performance using deep learning or hybrid approaches. However, a significant portion of these works rely on generic or benchmark datasets that do not fully capture the characteristics of smart home IoT traffic. In contrast, the proposed study utilizes

the COMNETS dataset, which better reflects real-world IoT environments, thereby providing more realistic evaluation conditions. Several important insights can be derived from the experimental results.

First, ensemble-based models consistently outperform single-tree approaches in terms of both classification performance and stability. Among all evaluated models, XGBoost achieves the best overall performance across all evaluation metrics. This result highlights the effectiveness of boosting-based methods in capturing complex and non-linear traffic patterns, particularly in imbalanced datasets commonly found in intrusion detection scenarios. Second, preprocessing plays a critical role in improving model performance. The integration of SMOTE and Min–Max normalization significantly enhances recall and F1-score, particularly for minority classes. This finding emphasizes that addressing data imbalance and feature scaling is essential for building reliable intrusion detection systems, as many traditional approaches tend to overlook minority attack classes. Third, the compact set of seven packet-level features keeps the models lightweight and interpretable, with no separate feature-selection stage required. A leakage-controlled re-evaluation, however, shows that the high headline scores are substantially inflated by evaluation artifacts and that honest performance is more modest. This is particularly important in IoT environments, where resource constraints require lightweight yet effective solutions.

Another key insight concerns the trade-off between performance and computational cost. While Decision Tree offers the fastest execution time, its lower accuracy and reduced robustness limit its applicability. Random Forest improves performance but incurs higher computational overhead. In contrast, XGBoost provides a balanced trade-off between accuracy and efficiency, making it a more practical choice for real-world deployment. Furthermore, the robustness analysis confirms that ensemble-based models are more resilient under noise-perturbed conditions. XGBoost, in particular, shows minimal performance degradation, indicating strong generalization capability in dynamic and potentially hostile IoT environments. Despite these promising results, several limitations should be acknowledged. First, although multiple datasets are utilized and cross-dataset validation is performed, the primary evaluation relies on the COMNETS dataset. This may still limit the generalization capability of the model across more diverse and heterogeneous IoT environments. Second, while noise-based perturbations are considered, more sophisticated attack strategies could be explored in future work to further assess model resilience. Overall, the findings demonstrate that a holistic framework integrating preprocessing and ensemble learning can significantly enhance intrusion detection performance while maintaining computational efficiency. These insights underline the importance of designing robust, scalable, and context-aware IDS solutions for smart home IoT environments.

4.11 Computational Complexity Analysis

To further evaluate the practicality of the proposed framework, computational complexity and runtime efficiency are analyzed. Let n denote the number of samples, d the number of features, and k the number of trees in ensemble models. For Decision Tree (DT), the training complexity is typically $O(n \cdot d \cdot \log n)$, as the algorithm recursively partitions the feature space based on impurity measures. While DT is computationally efficient and suitable for real-time applications, it is prone to overfitting, especially in high-dimensional IoT data.

Random Forest (RF) extends DT by constructing k independent trees using bootstrap sampling and random feature selection. Its overall training complexity can be approximated as $O(k \cdot n \cdot d \cdot \log n)$. Although RF improves generalization and robustness, the increased number of trees leads to higher computational and memory costs.

XGBoost, as a gradient boosting framework, builds trees sequentially. Its complexity can be expressed as $O(k \cdot n \cdot d \cdot \log n)$, with additional overhead due to gradient computation and regularization. However, XGBoost

incorporates several optimization strategies, such as parallel processing, tree pruning, and cache-aware computation, which significantly reduce practical runtime.

In contrast, deep learning models such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks require substantially higher computational resources due to multi-layer architectures and iterative backpropagation. LSTM, in particular, incurs additional computational cost due to sequential dependencies in temporal processing, limiting its suitability for real-time IoT environments. Overall, the results indicate that ensemble-based methods, particularly XGBoost, provide a favorable trade-off between computational efficiency and detection performance. This makes them more suitable for deployment in resource-constrained IoT systems, where both accuracy and efficiency are critical.

4.12 Evaluation Validity and Leakage Analysis

Because the high scores above were obtained under a conventional random split, we additionally examined the COMNETS capture for evaluation artifacts, as requested by the reviewers. The audit found three issues: about 26% of records are exact duplicates that can appear in both training and test partitions under a random split; the MITM and DoS classes originate almost entirely from a single source host; and the absolute-time field `frame.time_relative` is strongly associated with the label.

To quantify their impact, leakage controls were introduced cumulatively (duplicate removal, exclusion of `frame.time_relative`, and a temporally ordered split). Under the conventional protocol XGBoost reaches macro-F1 0.965; under the fully leakage-controlled protocol macro-F1 falls to 0.558 and benign detection essentially fails. The benign Average Precision drops from 0.951 to 0.008 even though its ROC-AUC stays at 0.996, which is why Precision–Recall curves (Fig. 7) are reported. On a Colab CPU, per-sample inference latency was 0.17 ms (DT), 1.78 ms (XGBoost), and 23.66 ms (RF), with model sizes of about 2.4, 6.1, and 84 MB. These results complement and contextualize the conventional-protocol numbers reported earlier.

An important consequence of the severe class imbalance in the COMNETS capture concerns the reliability of minority-class detection. Benign traffic constitutes only about 0.48% of the records, and under the temporally ordered split the test partition contains only 11 benign instances; with so few minority-class test samples, any estimate of benign recall or precision carries very wide statistical uncertainty. The collapse of the benign Average Precision from 0.951 to 0.008 under the leakage-controlled protocol further indicates that the apparently strong minority-class performance obtained under the conventional protocol is largely an evaluation artifact. Although SMOTE rebalances the training distribution, it cannot create new information about a class that is barely represented in the capture, and it has no effect on the statistical reliability of the test partition. We therefore explicitly caution that the ability of the evaluated models to reliably detect rare but critical traffic classes on this dataset remains unverified rather than demonstrated. Addressing this limitation requires longer benign-traffic capture campaigns, per-class reporting with confidence intervals, threshold-independent measures such as the area under the Precision–Recall curve, and complementary formulations such as cost-sensitive learning or anomaly-based (one-class) detection that do not depend on abundant minority-class labels; these directions are incorporated into the future work outlined in the conclusion.

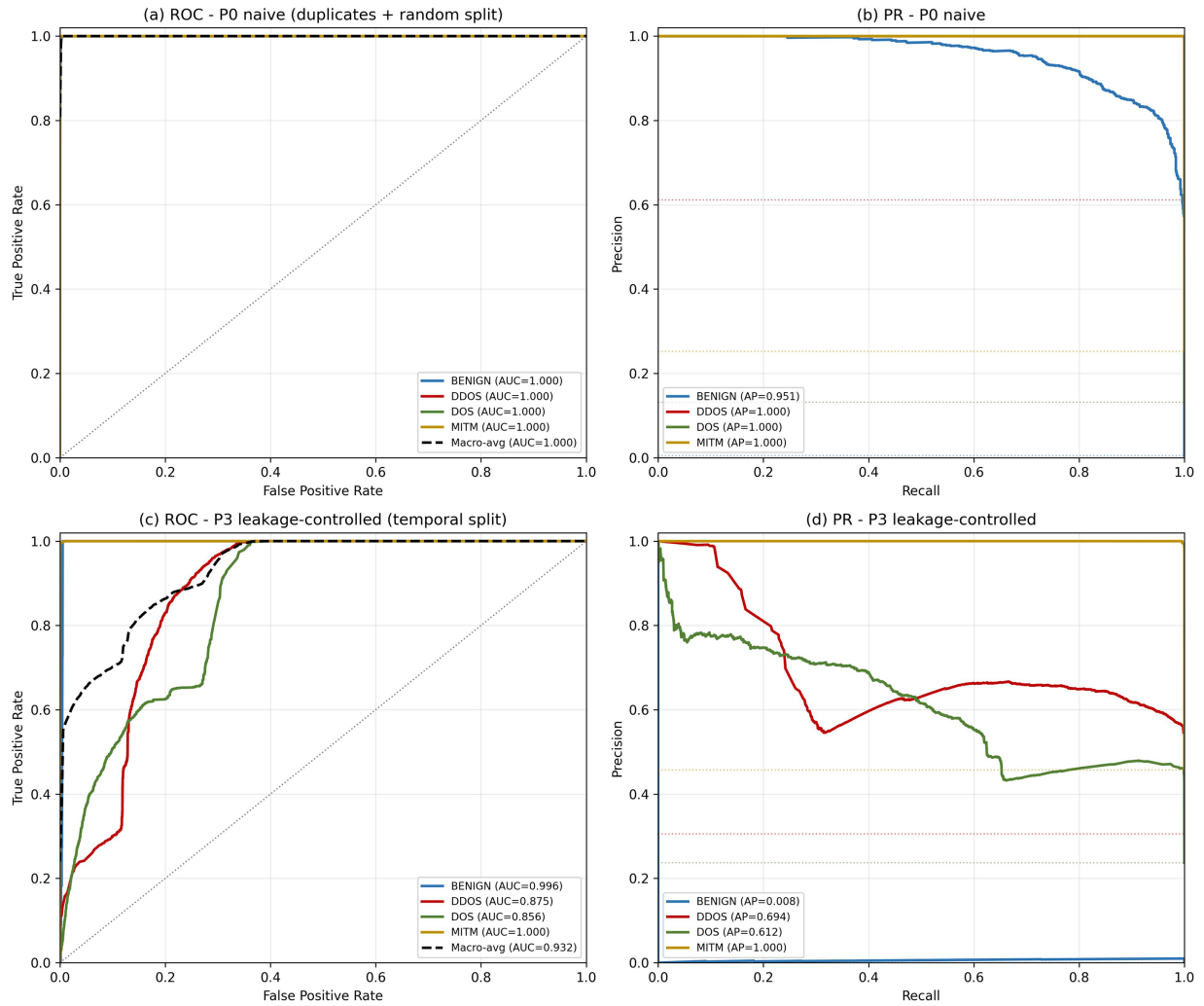


Figure 7: Comparison of ROC and Precision–Recall (PR) curves under the naive (P0) and leakage-controlled (P3) evaluation protocols: (a) ROC curves under the P0 naive protocol (duplicate records retained with a random split), where all classes and the macro-average reach AUC = 1.000; (b) PR curves under the P0 naive protocol, showing near-perfect average precision (BENIGN AP = 0.951; DDoS, DoS, and MITM AP = 1.000); (c) ROC curves under the P3 leakage-controlled protocol (temporally ordered split), where per-class AUC decreases (BENIGN = 0.996, DDoS = 0.875, DoS = 0.856, MITM = 1.000; macro-average = 0.932); and (d) PR curves under the P3 leakage-controlled protocol, where average precision drops sharply, most notably for the BENIGN class (AP = 0.008), with DDoS AP = 0.694, DoS AP = 0.612, and MITM AP = 1.000.

5 Conclusion

This study presents a robustness-aware intrusion detection framework designed for smart home IoT environments, addressing key challenges such as data imbalance, computational constraints, and model reliability under dynamic conditions. The proposed framework integrates preprocessing techniques and ensemble-based machine learning models into a unified and systematic pipeline. Under a conventional evaluation, ensemble-based approaches outperform single-model techniques, with XGBoost reaching about 97.8% accuracy; a leakage-controlled evaluation, however, shows this figure is largely an evaluation artifact (honest macro-F1 of 0.558), reported transparently here. Preprocessing such as SMOTE and Min–Max normalization is used to handle minority attack classes.

In addition, the robustness analysis shows that ensemble-based models are more resilient to noise-perturbed conditions. XGBoost exhibits the smallest performance degradation, indicating its ability to maintain stable detection capability under perturbed data. This highlights the importance of evaluating intrusion detection systems beyond standard accuracy metrics to ensure real-world applicability. Despite these promising results, several limitations remain. The evaluation is conducted using a single dataset, which may limit generalization across different IoT environments. Furthermore, the robustness analysis uses random Gaussian perturbation rather than gradient-based adversarial attacks (e.g., FGSM or PGD), so robustness to genuine adaptive attacks remains unverified. The evaluation relies on a single smart-home capture in which benign traffic is only about 0.48% of records; under a temporally ordered split the test partition contains only 11 benign records, so minority-class detection cannot be assessed reliably on this dataset. Consequently, the ability of the framework to detect rare but critical traffic classes should be regarded as an open question rather than an established capability, and the reported headline results should be interpreted accordingly. The evaluation is offline; on-device latency and memory on embedded gateways, and a valid cross-dataset study, are left for future work.

Future work will focus on extending the evaluation to multiple IoT datasets, collecting substantially larger volumes of benign smart-home traffic to enable statistically reliable minority-class evaluation, applying cost-sensitive and anomaly-based (one-class) detection formulations, and incorporating gradient-based adversarial attack simulations (e.g., FGSM and PGD) to further assess model robustness. Additionally, deploying the proposed framework in real-time IoT systems will be explored to validate its practical applicability. Overall, this study demonstrates that a holistic approach combining data preprocessing and ensemble learning can significantly enhance the performance and reliability of intrusion detection systems in smart home IoT environments.

Acknowledgement: The authors would like to express their sincere gratitude to Universitas Sriwijaya and Universitas Muhammadiyah Kotabumi for their institutional support. We also acknowledge the contributions of King Mongkut's University of Technology Thonburi (KMUTT), Universiti Teknologi Malaysia, and the National Research and Innovation Agency (BRIN) for their academic collaboration. Finally, we thank the providers of the COMNETS Smart Home datasets for supporting this research.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Adi Wibowo: Conceptualization, methodology, data curation, software, writing—original draft. Deris Stiawan: Supervision, validation, investigation, writing—review and editing. Chonlameth Arpnikanondt: Methodology, validation, writing—review and editing. Ahmad Heryanto: Formal analysis, methodology, investigation. Didi Rosiyadi: Data curation, validation, writing—review and editing. Mohd Yazid Idris: Supervision, writing—review and editing. Fandi Kurniawan: Investigation, methodology, writing—review and editing. Tuul Triyason: Validation, writing—review and editing. Rahmat Budiarto: Supervision, writing—review and editing. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data will be made available on request.

Ethics Approval: Not applicable.

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

References

1. Touqeer H, Zaman S, Amin R, Hussain M, Al-Turjman F, Bilal M. Smart home security: challenges, issues and solutions at different IoT layers. *J Supercomput*. 2021;77(12):14053–89. doi:10.1007/s11227-021-03825-1.

2. Siraparapu SR, Azad SMAK. Securing the IoT landscape: a comprehensive review of secure systems in the digital era. *e-Prime Adv Electr Eng Electron Energy*. 2024;10(2):100798. doi:10.1016/j.prime.2024.100798.
3. Bhardwaj A, Bharany S, Abulfaraj AW, Osman Ibrahim A, Nagmeldin W. Fortifying home IoT security: a framework for comprehensive examination of vulnerabilities and intrusion detection strategies for smart cities. *Egypt Inform J*. 2024;25(9):100443. doi:10.1016/j.eij.2024.100443.
4. Aldridge AA. Examining the security essences of Internet of Things (IoT) devices in smart homes: challenges, vulnerabilities, and countermeasures. *Issues Inf Syst*. 2024;25(1):279–92. doi:10.48009/1_iis_2024_123.
5. Alnasser O, Al Muhtadi J, Saleem K, Shrestha S. Signature and anomaly based intrusion detection system for secure IoTs and V2G communication. *Alex Eng J*. 2025;125(3):424–40. doi:10.1016/j.aej.2025.03.068.
6. Ahmad R, Alsmadi I. Machine learning approaches to IoT security: a systematic literature review. *Internet Things*. 2021;14(3):100365. doi:10.1016/j.iot.2021.100365.
7. Kantharaju V, Suresh H, Niranjnamurthy M, Ansarullah SI, Amin F, Alabrah A. Machine learning based intrusion detection framework for detecting security attacks in Internet of Things. *Sci Rep*. 2024;14(1):30275. doi:10.1038/s41598-024-81535-3.
8. Berhili M, Chaieb O, Benabdellah M. Intrusion detection systems in IoT based on machine learning: a state of the art. *Procedia Comput Sci*. 2024;251:99–107. doi:10.1016/j.procs.2024.11.089.
9. Rezvani S, Wang X. A broad review on class imbalance learning techniques. *Appl Soft Comput*. 2023;143(9):110415. doi:10.1016/j.asoc.2023.110415.
10. Chen W, Yang K, Yu Z, Shi Y, Chen CLP. A survey on imbalanced learning: latest research, applications and future directions. *Artif Intell Rev*. 2024;57(6):137. doi:10.1007/s10462-024-10759-6.
11. Ali Abdul Hassan H. Exploring lightweight deep learning techniques for intrusion detection systems in IoT networks: a survey. *J Electr Syst*. 2024;20(4s):1944–58. doi:10.52783/jes.2292.
12. Kivanç Eren K, Küçük K, Özyurt F, Alhazmi OH. Simple yet powerful: machine learning-based IoT intrusion system with smart preprocessing and feature generation rivals deep learning. *IEEE Access*. 2025;13(4):41435–55. doi:10.1109/ACCESS.2025.3547642.
13. Rejito J, Stiawan D, Alshaflut A, Budiarto R. Machine learning-based anomaly detection for smart home networks under adversarial attack. *Comput Sci Inf Technol*. 2024;5(2):122–9. doi:10.11591/csit.v5i2.pp122-129.
14. Apruzzese G, Andreolini M, Ferretti L, Marchetti M, Colajanni M. Modeling realistic adversarial attacks against network intrusion detection systems. *Digital Threats*. 2022;3(3):1–19. doi:10.1145/3469659.
15. Elreedy D, Atiya AF. A comprehensive analysis of synthetic minority oversampling technique (SMOTE) for handling class imbalance. *Inf Sci*. 2019;505(1):32–64. doi:10.1016/j.ins.2019.07.070.
16. Guezzaz A, Benkirane S, Azrou M, Khurram S. A reliable network intrusion detection approach using decision tree with enhanced data quality. *Secur Commun Netw*. 2021;2021(3):1230593. doi:10.1155/2021/1230593.
17. Ullah F, Ullah S, Srivastava G, Lin JC. IDS-INT: intrusion detection system using transformer-based transfer learning for imbalanced network traffic. *Digit Commun Netw*. 2024;10(1):190–204. doi:10.1016/j.dcan.2023.03.008.
18. Ibrahim Hairab B, Aslan HK, Said Elsayed M, Jurcut AD, Azer MA. Anomaly detection of zero-day attacks based on CNN and regularization techniques. *Electronics*. 2023;12(3):573. doi:10.3390/electronics12030573.
19. Chimphee W, Chimphee S. Hyperparameters optimization XGBoost for network intrusion detection using CSE-CIC-IDS, 2018 dataset. *IAES Int J Artif Intell*. 2024;13(1):817. doi:10.11591/ijai.v13.i1.pp817-826.
20. Hamidou ST, Mehdi A. Enhancing IDS performance through a comparative analysis of Random Forest, XGBoost, and Deep Neural Networks. *Mach Learn Appl*. 2025;22(12):100738. doi:10.1016/j.mlwa.2025.100738.
21. Sommer R, Paxson V. Outside the closed world: on using machine learning for network intrusion detection. In: *Proceedings of the 2010 IEEE Symposium on Security and Privacy*; 2010 May 16–19; Oakland, CA, USA. p. 305–16. doi:10.1109/sp.2010.25.
22. Apruzzese G, Laskov P, Schneider J. SoK: Pragmatic assessment of machine learning for network intrusion detection. In: *Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*; 2023 Jul 3–7; Delft, The Netherlands. p. 592–614. doi:10.1109/eurosp57164.2023.00042.
23. Kapoor S, Narayanan A. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*. 2023;4(9):100804. doi:10.1016/j.patter.2023.100804.

24. Engelen G, Rimmer V, Joosen W. Troubleshooting an intrusion detection dataset: the CICIDS2017 case study. In: Proceedings of the 2021 IEEE Security and Privacy Workshops (SPW); 2021 May 27; San Francisco, CA, USA. p. 7–12. doi:10.1109/spw53761.2021.00009.
25. Ring M, Wunderlich S, Scheuring D, Landes D, Hotho A. A survey of network-based intrusion detection data sets. *Comput Secur.* 2019;86(1):147–67. doi:10.1016/j.cose.2019.06.005.
26. Maseer ZK, Yusof R, Bahaman N, Mostafa SA, Foozy CFM. Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. *IEEE Access.* 2021;9:22351–70. doi:10.1109/ACCESS.2021.3056614.
27. Gerodimos A, Maglaras L, Ferrag MA, Ayres N, Kantzavelou I. IoT: communication protocols and security threats. *Internet Things Cyber Phys Syst.* 2023;3:1–13. doi:10.1016/j.iotcps.2022.12.003.
28. Pranav RP, Prawin RP, Paramasivan P, Shynu T, Subhashni R. IoT-SecNet: a trust-based framework for enhancing security in IoT communications and network layer protocols. *FMDB Trans Sustain Comput Syst.* 2024;2(2):96–106. doi:10.69888/fts.2024.000201.
29. Dash N, Chakravarty S, Rath AK, Giri NC, AboRas KM, Gowtham N. An optimized LSTM-based deep learning model for anomaly network intrusion detection. *Sci Rep.* 2025;15(1):1554. doi:10.1038/s41598-025-85248-z.
30. Koroniotis N, Moustafa N, Sitnikova E, Turnbull B. Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: bot-IoT dataset. *Future Gener Comput Syst.* 2019;100(7):779–96. doi:10.1016/j.future.2019.05.041.
31. Carvalho M, Pinho AJ, Bras S. Resampling approaches to handle class imbalance: a review from a data perspective. *J Big Data.* 2025;12(1):71. doi:10.1186/s40537-025-01119-4.
32. Garcia S, Parmisano A, Erquiaga MJ. IoT-23: a labeled dataset with malicious and benign IoT network traffic (Version 1.0.0) [Dataset]. Zenodo. 2020. doi:10.5281/zenodo.4743746.
33. Talukder MA, Sharmin S, Uddin MA, Islam MM, Aryal S. MLSTL-WSN: machine learning-based intrusion detection using SMOTETomek in WSNs. *Int J Inf Secur.* 2024;23(3):2139–58. doi:10.1007/s10207-024-00833-z.
34. Al Sawafi Y, Touzene A, Hedjam R. Hybrid deep learning-based intrusion detection system for RPL IoT networks. *J Sens Actuator Netw.* 2023;12(2):21. doi:10.3390/jsan12020021.
35. Choudhary S, Kesswani N. Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT. *Procedia Comput Sci.* 2020;167:1561–73. doi:10.1016/j.procs.2020.03.367.
36. Lo WW, Kulatilleke GK, Sarhan M, Layeghy S, Portmann M. XG-BoT: an explainable deep graph neural network for botnet detection and forensics. *Internet Things.* 2023;22(4):100747. doi:10.1016/j.iot.2023.100747.