University of Baghdad
College of Engineering
JOURNAL OF ENGINEERING

Journal of Engineering

journal homepage: www.jcoeng.edu.iq

Volume 32 Number 6 June 2026

Comparative Evaluation of Supervised Machine Learning Models for IoT Botnet Detection using Random Forest, XGBoost, and ANN

Ali Mohammed Noori Tarab

Department of Computer and Communications, Faculty of Engineering, Islamic University of Lebanon, Lebanon

ABSTRACT

This research presents a comparative experimental study of machine learning models for botnet attack detection in Internet of Things network using N-BaIoT dataset. The dataset consists of benign traffic and malicious traffic generated by Mirai and BASHLITE botnets family. A total of 115 traffic features are used for classification**.** The study compares three representative models of learning, namely Random Forest as a classical machine learning classifier, XGBoost as an advanced model of ensemble learning and fully connected Artificial Neural Network as a deep learning classifier. The evaluation of the models was done looking for performance metrics like accuracy, precision, recall, F1-score etc. As per the experiments conducted, the developed system achieved the best which is XGBoost which produced an accuracy rate of 99.12%, a precision of 98.98%, a 99.26% recall, 99.12% F1-score, a 0.88% false alarm rate, and an AUC which is 0.998. The model also achieved a remarkable inference time of 0.018 ms per sample, outperforming Random Forest with 0.021 ms and ANN at 0.035 ms. These results indicate that XGBoost provides the most effective balance between detection accuracy and computational efficiency, making it a suitable candidate for near-real-time IoT botnet detection at gateway or edge-monitoring levels.

Keywords: Botnet detection, Classical machine learning, Ensemble learning, Deep learning, Random forest, XGBoost, Cybersecurity.

1. INTRODUCTION

The Internet of Things (IoT) has become a building block of today's digital world, allowing connectivity spanning smart home applications, healthcare, industrial monitoring and automation, environmental monitoring, and intelligent transportation. In just a few years, the proliferation of connected devices has created enormous functional and economic benefits. However, it has expanded the cyber-attack surface like never before. IoT devices differ from traditional computing systems in that they lack computing power, memory, storage capacity, and embedded security controls; as a result, they can be compromised and abused remotely **(Lefoane et al., 2025; Gelgi et al., 2024).**

*Corresponding author
Peer review under the responsibility of University of Baghdad.
https://doi.org/10.31026/j.eng.2026.06.02

Article received: 27/03/2026
Article revised: 23/05/2026
Article accepted: 30/05/2026
Article published: 01/06/2026

One of the most disruptive security threats in this area is Botnet attacks, in which there is the covert recruitment and control of hundreds or thousands of infected devices in order to carry out malicious operations. Further, operations such as DDoS, Collection of reconnaissance, spam dissemination, credential theft and lateral propagation. The families of malware Mirai and BASHLITE proved that the IoT ecosystem could be weaponized at scale. It did so by causing these malware codes to use weak default credentials, outdated firmware and ill-secured service interfaces. Many IoT devices are deployed in unattended or low-maintenance environments so that this kind of compromise can go unnoticed for prolonged periods. As such, infected nodes may become a permanent part of the infrastructure of large-scale attacks **(Meidan et al., 2018; Gelgi et al., 2024).**

The use of traditional intrusion signature-based detection approaches as mitigation of IoT botnets is becoming inefficient as the attack traffic is dynamic, evasive, and behaviorally diverse. Static rule-based detection fails to generalize well across devices and also fails to adapt itself against varying attack patterns. This is especially true when the malicious traffic is obfuscated or not very different from normal traffic. Recent study focus has shifted towards artificial intelligence-based detection, especially machine learning and deep learning methods that are capable of capturing hidden statistical regularities in network traffic. Moreover, these methods can classify malicious behavior based on complex feature interactions **(Nazir et al., 2023; Lefoane et al., 2025).**

A significant amount of work has reported encouraging outcomes in the area of intelligentsia IoT botnet detection. A recent paper by **(Meidan et al., 2018)** proposed the N-BaIoT benchmark and demonstrated the feasibility of network-based anomaly detection using a deep autoencoder on traffic generated from the infection of commercial IoT devices. Subsequent research compared a classical and neural approach to the same problem. According to research conducted by **(Kim et al., 2020)** both the machine learning and deep learning techniques give strong detection performance on N-BaIoT. According to a different research, early detection at the network's Edge is practically important, in which latency and resource efficiency become the central design constraints **(Kumar et al., 2022)**. Recently, **(Malik et al., 2022)** suggested a lightweight one-class method and **(Wardana et al., 2024)** studied ensemble deep neural models for heterogeneous IoT devices, stating that high accuracy is no longer sufficient unless it is coupled with robustness across device diversity and realistic deployment settings.

At the same time, researchers are investigating productivity-enhancing strategies that facilitate the deployment of new technologies. Some methods, such as feature selection and dimensionality reduction, reduce redundancy, improve learning efficiency and reduce computing costs in IoT intrusion detection pipelines **(Liu and Du, 2023; Baker and Samarneh, 2024; Hossain and Islam, 2025).** Research on lightweight and quantized models has become popular in near-real-time detection applications for resource-constrained devices. As we optimize the model, there is a trade-off in its predictive quality and the cost of inference and memory size **(Khan et al., 2025).** The emergence of Explainable Artificial Intelligence is an important complementary direction that reflects the need for security models that are not only accurate but also interpretable and operationally trusted **(Saied and Guirguis, 2025).**

There are still issues that remain in published literature. Several works stress accuracy while putting less emphasis on false alarm rate, computational cost, and detection time, even though these factors have a strong association with near-real-time applicability in limited IoT environments. Second, some works are focused on a single model family, so it is difficult

to tell whether the reported gains come either from modifications to the learning algorithm or from post hoc differences in preprocessing, feature engineering, or evaluation protocols. Recent studies highlight the relevance of feature tuning. However, there is a lack of consensus on which of the model families can give the best predictive accuracy and operational efficiency in a unified experimental setting **(Liu and Du, 2023; Baker and Samarneh, 2024; Lefoane et al., 2025).**

A systematic empirical evaluation is needed for representative models from different intelligent learning paradigms in a common setting over a realistic IoT dataset. This study involves three models for comparison and assessment. The first model is a random forest, which is part of classical machine learning. This is robust. The second method is XGBoost, which is a modern one. Ensemble learning proves to be very efficient in the case of structured tabular data. Artificial Neural Networks (ANN) are a representative of deep learning models, which is the third type. The chosen models differ in their learning philosophy, which enables us to perform a grounded comparison of interpretability, predictive power and computational efficiency in the same detecting context. The selected benchmark N-BaIoT dataset for the experimentation represents real benign and malicious traffic originating from commercial devices which infect Mirai and BASHLITE. Therefore, the N-BaIoT dataset can be a benchmark for botnet detection **(Meidan et al., 2018; Kim et al., 2020).**

The importance of this initiative is the transition from writing on single models to evaluation, which considers detection performance and operational aptness for deployment in actual IoT security settings. The present study analyzes model behavior based on the Accuracy, Recall, False Alarm Rate (FAR) and the detection time to choose the model that provides the best trade-off between prediction and execution. According to the research, the paper also contributes to literature by offering a comparative study of three well-known intelligent models utilizing the same dataset, preprocessing, and evaluation conditions. This will help design more robust and practically deployable intelligent intrusion detection schemes for resource-limited IoT networks.

According to recent directions and research, scientists consider IoT botnet detection as one of the major challenges in cybersecurity. Due to dynamic and heterogeneous attack traffic in the IoT environment, conventional signature-based methods are less effective. As demonstrated by the publication of recent reviews, machine learning or deep learning-based approaches to IoT botnets detection will gain importance over time, especially if it is to be used in resource-constrained environments, which can readily adapt intelligently to potential threats **(Nazir et al., 2023; Lefoane et al., 2025).**

A significant advancement in this area was the establishment of the N-BaIoT dataset, which offered a realistic benchmark with benign and malicious activity gathered from commercial IoT devices compromised by Mirai and BASHLITE malware. N-BaIoT is one of the widely used datasets intended for evaluating intelligent botnet detection models. Furthermore, N-BaIoT has shown that a machine learning and deep learning technique can perform sufficiently on the dataset **(Meidan et al., 2018; Kim et al., 2020).**

A recent study has shifted from merely providing the detection precision of the raw model to features that enhance the computational efficiency, deployment suitability and interpretability of the models. Ensemble approaches that are based on feature selection reduce input dimensionality to improve detection performance. Lightweight optimized models like quantized XGBoost lower inference cost without affecting classification performance in the context of resource-limited IoT. Simultaneously, Explainable AI became

an important direction, which shows that botnet detection systems are referring to interpretability besides functionality traction **(Hossain and Islam, 2025; Khan et al., 2025; Saied and Guirguis, 2025).**

To give a clearer insight into how the recent studies assessed the IoT botnet detection over N-BaIoT, **Table 1** presents a summary of the representative works in terms of dataset, evaluated algorithms, reported performance metrics, and principal limitations.

Table 1. A comparative summary of representative IoT botnet detection studies using the N-BaIoT dataset.

Study	Dataset	Algorithm(s)	Accuracy (%)	F1-Score (%)	FAR (%)	Main limitation
(Wardana et al., 2024)	N-BaIoT	Ensemble averaging DNN	97.21	88.48	Not reported	Evaluation limited to N-BaIoT
(Hossain and Islam, 2025)	N-BaIoT	Feature selection-driven ensemble learning using SVM, LR, KNN, NB, DT, and RF	99.00 (RF)	79.00 (RF)	Not reported	Some models, especially SVM and LR, showed weaker performance on nonlinear traffic; FAR was not reported
(Wu and Chen, 2022)	N-BaIoT	Deep forest	99.98	Not reported	Not reported	Limited evidence of cross-dataset validation
(Abbasi et al., 2021)	N-BaIoT	Logistic Regression, ANN, Autoencoders	99.98	Not reported	Not reported	Limited discussion of generalization and deployment-related constraints
(Catillo et al., 2023)	N-BaIoT	Deep autoencoder, semi-supervised	Up to 99.99	99.96–99.98	0.00–0.71	Requires representative normal traffic for training and reports device-specific evaluation
(Cao et al., 2023)	N-BaIoT	Feature selection + TPE-LightGBM ensemble	99.97	Not reported	Not reported	Experiments reported only on N-BaIoT
(Alharbi et al., 2021)	N-BaIoT	LGBA-NN	90.00	Not reported	Not reported	Lower performance than several competing approaches
(Hashim et al., 2024)	N-BaIoT	CNN-BiGRU-BiLSTM hybrid	98.87	Not reported	Not reported	Limited evidence of validation beyond N-BaIoT
(Cunha et al., 2022)	N-BaIoT and Bot-IoT	CNN	100.00	100.00	Not reported	Generalization limits not clearly discussed
(Serhane et al., 2025)	N-BaIoT	PCA + Extra Trees	99.94	Not reported	Not reported	FAR and cross-dataset robustness not sufficiently reported

As reflected in **Table 1,** the majority of past academic efforts achieved accurate and close to accurate performance on N-BaIoT. However, there are certain limitations which the literature shows. To begin with, performance reporting is highly inconsistent, as many studies focus on reporting accuracy only, omitting the F1-score, false alarm rate or class-level behavior in full detail. Next, very few studies report metrics that are directly relevant for deployment, specifically FAR and computational efficiency. Many evaluations are still conducted solely on N-BaIoT, questioning claims of real-world generalizability. Depending on benchmark or device-specific setting, the effect of this print is generally with the most reliable performance.

Research indicates that the industry is transitioning toward detection effectiveness computation efficiency and real-world deploying ability beyond just accuracy. There are fewer studies with a unified comparative framework which compares Random Forest, XGBoost and ANN altogether on a real-life benchmark like N-BaIoT while considering detection-related measures and operational indicators like false alarm rate and detection time. As per the existing gap, there is scope for a well-balanced comparative experimental study that will finally help in identifying the most applicable intelligent model to detect IoT botnet in real time **(Lefoane et al., 2025; Hossain and Islam, 2025; Khan et al., 2025).**

2. METHODOLOGY

2.1 Experimental Design

This research utilized a comparative experimental design to compare the efficacy of artificial intelligence models in detecting botnet attacks in IoT networks. The experimental design guarantees that all selected models use the same data preparation and evaluation protocol for fairness, reproducibility and practical relevance. The experiment deals with the binary classification which classifies the network traffic records in two classes botnet attack and benign traffic. The rationale for selecting this design is that early identification of normal and malicious traffic is the first and most essential step of practical IoT botnet defense systems **Fig. 1** ahows the overall workings of the proposed experimental framework.

A comparative evaluation of intelligent learning models: an experimental framework. The model begins with Random Forest (RF), which is selected as a classic. The second one is Extreme Gradient Boosting (XGBoost). The ensemble learning model was selected due to its power on the structured tabular data. The third model is an artificial neural network (ANN). This is a typical deep learning mechanism suited for nonlinear pattern learning. The use of these three models helps to find out whether boosting-based ensemble learning offers superior predictive accuracy and computational efficiency in comparison with bagging-based learning and neural classification.

The design of the experiment is organized in a single pipeline, allowing for a controlled and reproducible comparison of results. The five stages of the experiment are: dataset construction, data preparation, model training, model evaluation and comparative analysis. The use of this single pipeline guarantees that the distinctions observed with respect to the Random Forest, XGBoost and ANN are due to the learning behavior of the models and not due to dissimilarities in data handling, preprocessing or evaluation. As presented in Figure 1, the N-BaIoT dataset is selected to start the process since it is a well-known benchmark for IoT botnet detection. Moreover, it contains real benign and malicious traffic generated by commercial IoT devices. The malicious traffic of the dataset is implanted by first infecting various commercial IoT devices with Mirai and BASHLITE botnets **(Meidan et al., 2018;**

Kim et al., 2020). The first step is to select and label traffic samples to create a supervised classification corpus. The second stage of this process involves performing preprocessing operations such as data cleaning, label encoding, normalization, etc., to improve the consistency of data to assist stable model learning. Preprocessing is essential as it is likely that the IoT network traffic features differ in scale and statistical distribution and it may affect the performance of the classifier if they are not processed similarly **(Nazir et al., 2023).** During the third stage, all models were trained using the same stratified train–test partitioning protocol to maintain class distribution and limit evaluation bias. In the fourth stage, the trained models are evaluated using the same predictive and operational metrics, including accuracy, recall, false alarm rate, AUC, and detection time. These metrics were selected because IoT intrusion detection systems require not only high classification accuracy but also low false alarms and fast inference for practical deployment in resource-constrained environments **(Kumar et al., 2022; Lefoane et al., 2025).** Finally, the fifth stage compares the evaluated models to identify the classifier that provides the best trade-off between detection effectiveness and computational cost.

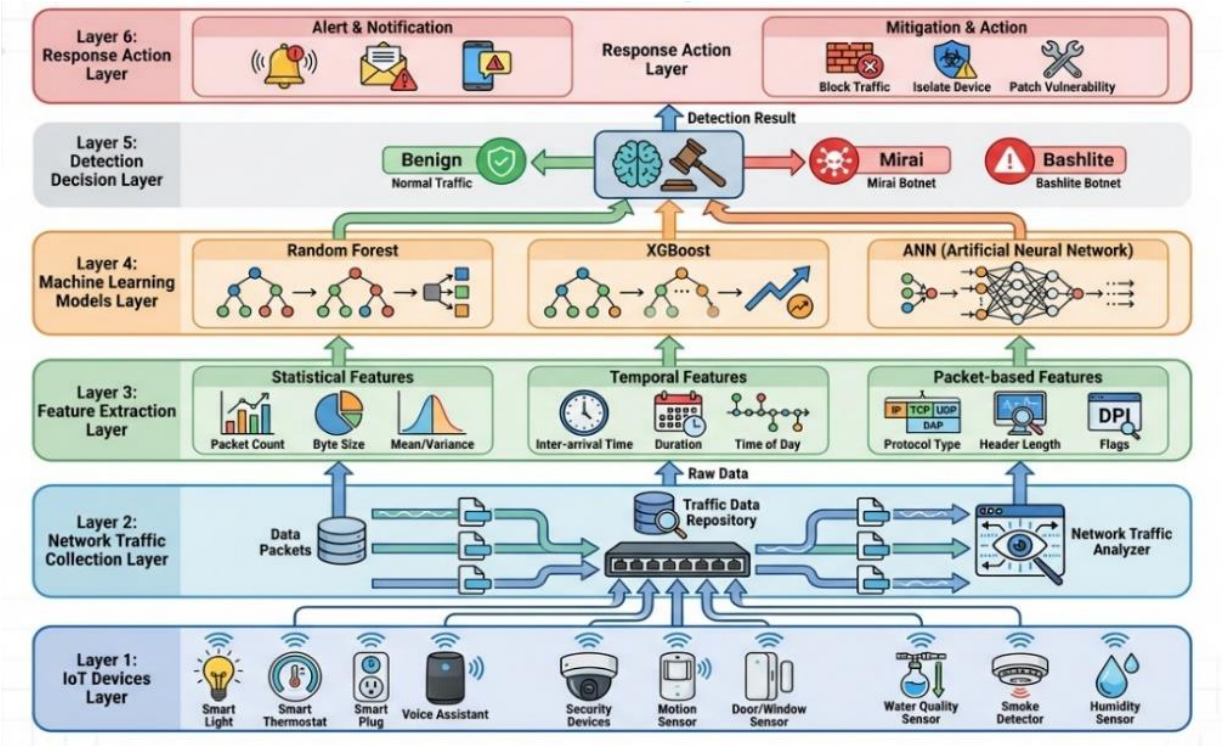

Figure 1. Proposed Experimental Framework for IoT Botnet Detection.

The stratified data-splitting design was employed in order to keep the distribution in check in train and test. As per our experiments, the model behavior of a priori could be affected due to biased partitioning. Additionally, the tuning of hyperparameters takes place independently for each classifier, and in the same experimental environment, in order that the comparisons are of model capability, not different tuning of the same model. Ultimately, the analysis assessment relies more on efficiency-oriented metrics than on classification-oriented metrics. Thus, this study not only checks whether a model can classify botnet traffic accurately but also whether the model can do so with sufficiently low false alarms and acceptable computational cost for IoT.

2.2 Dataset Construction and Description

The experiments conducted in this paper are based on the N-BaIoT dataset, a widely used and realistic benchmark for IoT botnet detection **(Table 2)**. The dataset contains network traffic collected from commercial IoT devices under normal operating conditions, as well as traffic captured after the devices were infected with the Mirai and BASHLITE botnet families. In the present study, the data were organized within a binary classification framework, where benign traffic represents the normal class and both Mirai and BASHLITE traffic represent the malicious attack class. However, the attack-family information was retained for descriptive and class-wise analysis.

A stratified subset of 100,000 samples was constructed from the N-BaIoT dataset. The subset included 50,000 benign samples and 50,000 malicious samples, distributed equally between Mirai and BASHLITE attacks. Each sample was represented by 115 numerical traffic features extracted from temporal traffic behaviour, packet statistics, and communication patterns. The use of this balanced and feature-complete subset allowed the evaluated models to be compared under the same experimental conditions.

No separate feature selection method was applied in the present study because the main objective was to compare Random Forest, XGBoost, and ANN under a unified baseline using the complete 115-feature representation of the N-BaIoT dataset. Using the full feature set allowed the models to learn from the complete temporal and statistical traffic profile without introducing bias from a model-specific feature-selection procedure. However, feature-importance analysis was conducted after training to identify the most influential variables. Future work may investigate optimized feature-selection methods to reduce dimensionality and computational cost.

Table 2. Description of the experimental data used in this study.

Dataset Source	Device Category	Traffic Type	Attack Family	Class Label	Number of Samples	Number of Features	Role in Experiment
N-BaIoT	Commercial IoT devices	Benign traffic	None	Benign	50,000	115	Reference normal class
N-BaIoT	Commercial IoT devices	Malicious traffic	Mirai	Attack	25,000	115	Positive attack class
N-BaIoT	Commercial IoT devices	Malicious traffic	BASHLITE	Attack	25,000	115	Positive attack class
Total	—	—	—	—	100,000	115	Complete experimental subset

2.3 Data Processing

Before the model training, the dataset was submitted to a preprocessing stage to the data quality, ensuring the consistency of data and allowing a fair comparison among the models being evaluated. This stage was an important one since the raw traffic data of IoT relates to missing values, duplicated records, scaling differences, and class-distribution issues. The presence of these issues impacts classifier performance and the reliability of experimental results.

Preprocessing phase started with inspecting the data. The chosen records were examined for missing values, duplicates, inconsistencies, and possible class imbalance. Surplus records

were taken out to avoid redundancy and ensure that identical traffic patterns do not skew training and evaluation. Subsequently, the target labels underwent encoding to numerical format for supervised classification Traffic produced by Mirai and BASHLITE botnets is regarded as malicious attack traffic, while benign traffic is referred to as normal class.

To prevent data leakage, the dataset was initially stratified split into training, validation and testing subsets. To maintain the proportion of classes in all sets and reduce biases in evaluations, stratified splitting was performed. Feature normalization was done after splitting and not before it. To be more specific, the Standard Scaler was only fitted on the training subset so that the scaling parameters learned from the training data could then be applied to the validation and testing subsets. This process makes sure that the model is not trained on any information from either the validation data or the test data, and a model's estimate will thus be unbiased. This amendment is consistent with the suggested process to avert data leaks during machine learning **(Kaufman et al., 2012).**

The normalization of features was carried out to mitigate the scale variation amongst the numerical attributes and also to improve the learning process. This was particularly essential for Artificial Neural Network (ANN), which is sensitive to the scale of features. Models like Random Forest and XGBoost, which are tree-based, are not sensitive to the issue of scaling of features, but in order to maintain a methodology, all models were provided with the same preprocessing pipeline. If class balancing was required, it was applied only to the training subset after the train–validation–test split, while the validation and testing subsets were kept unchanged to preserve an unbiased evaluation setting.

Thus, the corrected preprocessing sequence used in this study **(Table 3)** was as follows: data inspection, duplicate and missing-value handling, label encoding, stratified train–validation–test splitting, fitting the Standard Scaler on the training subset only, transforming the validation and testing subsets using the training scaler, optional training-set balancing, model training, and final evaluation

Table 3. Procedures used to preprocess the data in this study

Step	Operation	Purpose
1	Data inspection	To identify missing values, duplicated records, inconsistencies, and class-distribution issues in the selected traffic data
2	Missing-value and duplicate handling	To remove incomplete or redundant records and improve the quality of the dataset
3	Label encoding	To convert benign and attack traffic classes into numerical labels for binary classification
4	Stratified train–test splitting	To preserve class distribution in the training and testing subsets before applying any data-driven transformation
5	Feature normalization	To fit the StandardScaler on the training subset only and then apply the learned scaling parameters to the testing subset
6	Data balancing, if required	To reduce model bias caused by class imbalance, this step is applied only to the training subset
7	Model-ready dataset preparation	To provide consistent and leakage-free input data for Random Forest, XGBoost, and ANN evaluation

2.4 Model Development and Implementation

In this study, three classification models were developed and implemented so as to provide a balanced comparison among different artificial intelligence paradigms for IoT botnet

detection. The selected models include Random Forest (RF), Extreme Gradient Boosting (XGBoost) and Artificial Neural Network (ANN). The above models were selected as they appeared to be the three distinct yet popular approaches for learning from structured traffic data, bagging-based ensemble learning, boosting-based ensemble learning and nonlinear neural learning, respectively. Model selection is also in line with the goal of the study, which is to identify the model that offers the best trade-off between predictive effectiveness and computational efficiency under resource constraints in IoT environments. As depicted in **Fig. 2**, the model development and implementation workflow is as a whole.

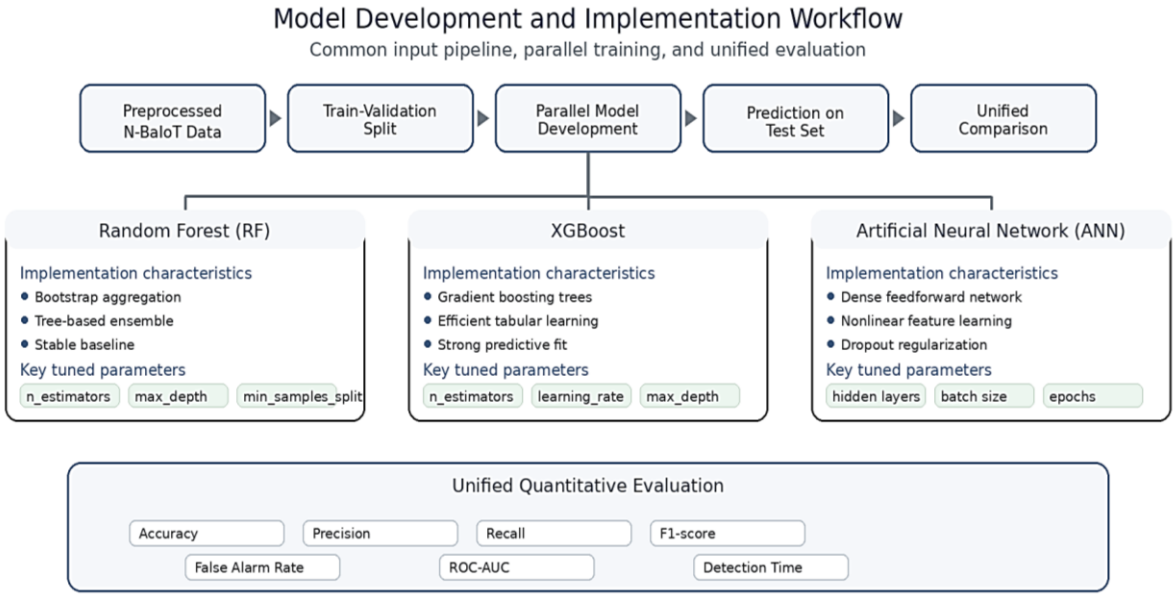

Figure 2. Model development and implementation workflow of the proposed study.

Implemented Random Forest as a supervised ensemble classifier built with multiple decision trees using bootstrap samples of the pre-processed dataset. Each tree gave a separate prediction, and then the final class was decided by a majority vote. The Random Forest classifier was chosen as it has a proven ability to enclose overfitting and tends to perform well on tabular data. It is essentially an excellent benchmark for evaluating how well network traffic can be classified. The RF model has three main parameters which are tuned: number of trees; maximum depth of trees; minimum number of samples for splitting a node. To maintain calculation stability, we adjust the above parameters for a better distinction between benign and malicious traffic.

The second model that was implemented was XGBoost which is an optimized gradient boosting classifier where trees are trained sequentially and each new tree tries to reduce the residual error of the previous ensemble. For our model we chose XGBoost as it is efficient in structured numerical data, supports regularization and is powerful predictor with relatively efficient inference. Consequently, it is particularly well-suited for IoT intrusion-detection problems where both accuracy and runtime are important. According to Present Study, the Number of Estimators of learning rate, maximum tree depth, subsampling ratio, and column sampling ratio are selection hyperparameters of XGBoost. In order to ensure comparison fairness, these settings were tuned using the same data conditions used for the other models. The third model, designed for binary classification of normalized traffic features, was the Artificial Neural Network, which has a fully connected feedforward architecture. The input

layer of ANN consists of the selected features, followed by a few hidden dense layers that contain rectified linear unit activation function and dropout regularization to prevent overfitting. The sigmoid activation was used on the output layer to predict the attack probability. The ANN model, which used binary cross-entropy loss, was trained via supervised binary classification tasks using the Adam optimizer. This model is selected due to the fact it's a nonlinear deep learning model which enables it to capture the difficult relationship between traffic features, which tree-based ensembles cannot do.

In order to maintain methodological consistency, all three models were evaluated with the same preprocessing pipeline and the binary class definition adopted in the study. Using the previously preprocessed N-BaIoT dataset, the process started with performing train-validation partitioning and proceeded to tune model-specific parameters and test. The models are subsequently evaluated in a harmonized evaluation framework based on predictive and operational performance metrics. By using a common implementation protocol, the research mitigates methodological bias and allows the observed differences in performance to be attributed to model behavior rather than differences in data handling or evaluation design. **Table 4** summarizes the implementation structure of the three models and their mutual evaluation pipeline. **Fig. 2** illustrates the structure.

Table 4. Setup of the investigated models in the proposed research.

Model	**Learning Paradigm**	**Core Structure**	**Main Tuned Parameters**	**Implementation Purpose**
Random Forest (RF)	Bagging-based ensemble learning	Multiple decision trees with majority voting	n_estimators, max_depth, min_samples_split, min_samples_leaf	Baseline robust classifier for structured traffic data
XGBoost	Boosting-based ensemble learning	Sequential gradient-boosted decision trees	n_estimators, learning_rate, max_depth, subsample, colsample_bytree	High-performance ensemble model for efficient IoT botnet detection
Artificial Neural Network (ANN)	Deep learning	Fully connected feedforward neural network	hidden layers, neurons per layer, batch size, epochs, dropout rate	Nonlinear pattern learning for binary traffic classification

2.5 Hyperparameter Settings and Experimental Environment

To enable a fair, reproducible and technically consistent comparison of the evaluated models, they were trained with hyperparameter settings that were selected according to the nature of structured IoT traffic data and binary classification task adopted in this study. The intention behind this phase was not to generate the greatest results through arbitrary adjustments to the model. However, we intended to achieve a level of balance that allowed for a fair comparison between Random Forest, XGBoost and ANN in terms of prediction and operation.

Because the three evaluated models belong to different learning paradigms, its configuration was determined according to model-specific tuning. In the Random Forest case, the settings adopted were aimed at controlling the ensemble size and complexity of trees for a baseline classifier with robustness but also for constraint overfitting. The chosen parameters for XGBoost were to achieve convergence behavior, generalization and computational efficiency during boosting-based learning. The selected architecture and training settings for the ANN

model were chosen to provide sufficient nonlinear learning capacity while sustaining optimization stability and reducing overfitting with dropout regularization. It can be seen from **Fig. 3** that in the current study, hyperparameter tuning was performed by a model-specific configuration phase done before choosing the final adopted settings for comparison.

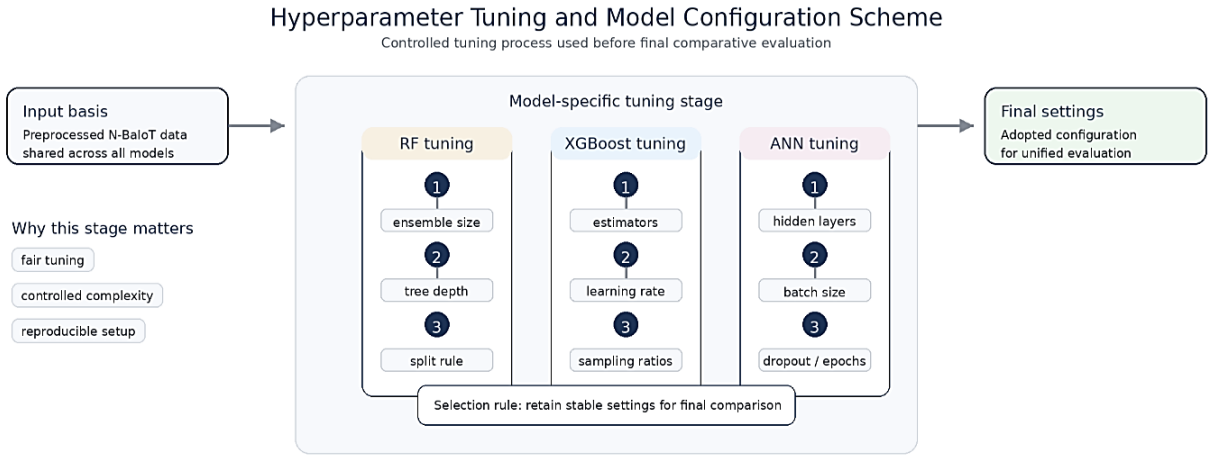

Figure 3. The RF, XGBoost, and ANN hyperparameter tuning and configuration scheme used before final model evaluation.

Table 5 summarizes the final hyperparameter settings adopted for the three evaluated models. These settings were selected to create a stable and reproducible training framework, rather than an overly aggressive optimization procedure. Since this study aims at a comparative evaluation under equivalent experimental conditions, each model is set for its own learning mechanism.

Table 5. Hyperparameters settings of the evaluated models used for training.

Model	Hyperparameter	Value	Model	Hyperparameter	Value
Random Forest	n_estimators	200	ANN	Hidden layers	3
Random Forest	max_depth	30	ANN	Neurons per hidden layer	128, 64, 32
Random Forest	min_samples_split	2	ANN	Activation function	ReLU
Random Forest	min_samples_leaf	1	ANN	Output layer	1 neuron
Random Forest	random_state	42	ANN	Output activation	Sigmoid
XGBoost	n_estimators	200	ANN	Loss function	Binary cross-entropy
XGBoost	learning_rate	0.10	ANN	Batch size	32
XGBoost	max_depth	5	ANN	Epochs	50
XGBoost	subsample	0.80	ANN	Optimizer	Adam
XGBoost	colsample_bytree	0.90	ANN	Learning rate	0.001
XGBoost	random_state	42	ANN	Dropout rate	0.30
ANN	Architecture	Fully connected feedforward ANN	ANN	Early stopping patience	10
ANN	Input layer	115 neurons			

The configuration for the Random Forest focused on ensemble size and tree-splitting behavior. The parameters relating to boosting including learning rate, depth of the tree and sampling ratios, etc. were configured for XGBoost. The ANN model architecture and optimizers were configured by choosing a number of hidden layers and neurons per layer, dropout rate, batch size, epochs, optimizers, and loss function. The parameterization strategy enhances methodological fairness because all models are evaluated using settings which are consistent with their internal structures. At the same time, parameterization restricts the training and testing of all models to the same experimental pipeline.
Further, to minimize implementation bias, all experiments were implemented in the same software and hardware environments. The training time, inference time, and reproducibility of deep learning models can differ due to various factors. As a result, similar computational settings were used for Random Forest, XGBoost and ANN. As shown in **Table 6**, software tools and hardware specifications in this study.

Table 6. A software and hardware testbed.

Item	**Specification**
Programming language	Python 3.x
Data handling library	Pandas
Numerical computation	NumPy
Machine learning library	Scikit-learn
Boosting library	XGBoost
Deep learning library	TensorFlow / Keras
Development environment	Jupyter Notebook or Google Colab
Processor	Intel Core i7 or equivalent
RAM	16 GB
GPU	Optional, if available
Operating system	Windows or Linux-based environment

2.6 Performance Evaluation Metrics

To evaluate the performance of the assessed models, we utilized a series of quantifiable metrics that measure both classification efficiency and practical effectiveness for IoT botnet detection. An efficient intrusion detection model must not only achieve high accuracy but also high attack-detection capability, low false alert behavior, and low inference latency in resource-constrained IoT settings. Accordingly, accuracy; precision; recall; F1-score; false alarm rate; detection time; and ROC-AUC and significant testing were the main evaluation criteria of this work. Metrics like accuracy, precision, recall, and f-score are widely used in classification and intrusion detection studies because they provide complementary information on how correct the predictions are, how erroneous the behavior is and how efficient the model is in terms of performance and operational cost **(Fawcett, 2006; Sokolova and Lapalme, 2009; Powers, 2011).**
Accuracy measures the proportion of traffic samples that are classified correctly among all tested samples. It gives a general indication of the correctness of the classification but may not be sufficient when the cost of false positive and false negative is not the same, like in cybersecurity application **(Sokolova and Lapalme, 2009; Powers, 2011).** Accuracy is calculated as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

where TP denotes true positives, TN denotes true negatives, FP denotes false positives, and FN denotes false negatives.

Precision indicates how many samples classified as attacks are indeed malicious. In the context of IoT intrusion detection, precision refers to the number of false positives rather than false negatives. Low precision means a large number of false alarms which may reduce trust in the detection system and the administrative responses. Furthermore, to handle false alarms, unnecessary administrative resources are being wasted which consumes network bandwidth as well **(Sokolova and Lapalme, 2009; Powers, 2011).** Precision is calculated as follows:

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall, also referred to as detection rate or sensitivity, measures the ability of the model to detect malicious traffic. For botnet detection, one wants high recall as missed attacks may allow compromised IoT devices to stay inside the network **(Powers, 2011).** Recall is calculated as follows:

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

The F1-score gives a sensible compromise between accuracy and recall. This tool is especially useful when false positives and false negatives are both important to evaluate the performance of a classification model **(Sokolova and Lapalme, 2009; Powers, 2011).** The F1-score is calculated as follows:

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (4)$$

The False Alarm Rate refers to the ratio of benign samples that have been wrongly classified as attacks. Operational measurement is important in IoT security because too many false alarms can burden security administrators, overload computational resources, and diminish confidence in the detection mechanism (**Powers, 2011).** FAR is calculated as follows:

$$FAR = \frac{FP}{FP+TN} \quad (5)$$

Detection time was utilized for assessing evaluation efficiency. It indicates how long a trained model takes to classify one test sample on average. The low inference latency is essential for gateway or edge-monitoring level near-real-time deployment **(Kumar et al., 2022).** Detection time is calculated as follows:

$$Detection\ time = \frac{T_{total}}{N} \quad (6)$$

where T_{total} denotes the total prediction time over the test set, and N denotes the number of tested samples.

Besides, Receiver Operating Characteristic analysis and the Area Under the Curve were used to assess threshold-independent discrimination capacity. The ROC analysis looks at the true positive rate and false positive rate at different classification thresholds, while the AUC summarizes the overall separability of the benign and malicious traffic classes. When AUC is high, it indicates strong discrimination power **(Fawcett, 2006).**

The differences in performance numbers between the models tested were tiny so statistical significance testing was included in the analysis. By employing McNemar's test, the paired classification results of XGBoost and the other models on the same test samples were compared. The test is suitable for assessing supervised classifiers while both classifiers make predictions on the same test instances. Furthermore, the test uses only discordant cases when one classifier is correct, and the other is not **(Dietterich, 1998).** McNemar's test statistics are calculated as follows:

$$\chi^2 = \frac{(|b-c|-1)^2}{b+c} \quad (7)$$

where b represents the number of misclassified samples of the first model and classified correctly by the second model, while c represents the number of samples classified correctly by the first model and misclassified by the second model. The significance level was taken as 0.05 Thus, the advantage of XGBoost was understood not only from small numerical differences in accuracy but also from a statistical comparison of paired model predictions.
The models that were evaluated used predictive and operational measures described earlier. The significance of this evaluation methodology is that it does not allow for the choice of the most suitable IoT botnet detection model based only on accuracy. The goal of the solution is not to achieve the perfect balance but rather a robust trade-off between successful attack detection, low false alarm, and inference performance following realistic IoT scenarios. As a result, the chosen evaluation framework accounted for classification effectiveness and computational efficiency in order to select the model that is more appropriate for near-real-time IoT botnet detection at gateway or edge-monitoring levels.

3. RESULTS AND DISCUSSION

This section includes the experimental results following the comparative analysis of Random Forest, XGBoost, and Artificial Neural Networks for the detection of Botnet attacks in IoT using the N-BaIoT dataset. The results are illustrated by effectiveness in classification, behavior for detection at class-level, efficiency in computation, and relative contribution of the most informative traffic features.

3.1 Experimental Configuration

The experiments were conducted on a workstation equipped with an Intel Core i7-10700K CPU with 8 cores at 3.8 GHz, 32 GB RAM, and an NVIDIA RTX 3060 GPU. The models were implemented using Python 3.9, scikit-learn v1.0.2, XGBoost v1.5.0, and TensorFlow v2.8.0.
The experimental data were derived from the N-BaIoT dataset introduced by **(Meidan, 2018)**. The original N-BaIoT dataset contains real network traffic collected from nine commercial IoT devices infected with the Mirai and BASHLITE botnet families. These devices include doorbells, a thermostat, a baby monitor, security cameras, and a webcam. The dataset provides traffic records represented by 115 numerical features extracted from temporal traffic behavior, packet statistics, and communication patterns.
Since the full N-BaIoT dataset contains more than seven million records, a stratified device-aware subset of 100,000 samples was constructed for the present comparative experiment. The subset was not taken from a single device. In contrast, samples from all nine device-specific datasets were randomly drawn while preserving representation for benign traffic, Mirai traffic, and BASHLITE traffic. By adopting this sampling strategy, we intend to avoid any device type bias and sufficiently test the evaluated models on heterogeneous IoT traffic.

The adopted subset contained 50,000 benign samples, 25,000 Mirai samples, and 25,000 BASHLITE samples. The ratio was selected to provide a realistic benign-traffic presence while maintaining equal representation between the two botnet families. Equal attack-family representation was important because the objective of the study was to compare the detection behaviour of Random Forest, XGBoost, and ANN under a fair experimental setting rather than allowing one attack family to dominate the learning process. The device-level composition of the selected subset is shown in **Table 7.**

Table 7. Device-level composition of the selected N-BaIoT subset.

Device	Device category	Benign samples	Mirai samples	BASHLITE samples	Total samples
Danmini	Doorbell	5,556	2,778	2,778	11,112
Ennio	Doorbell	5,556	2,778	2,778	11,112
Ecobee	Thermostat	5,556	2,778	2,778	11,112
Philips B120N/10	Baby monitor	5,556	2,778	2,778	11,112
Provision PT-737E	Security camera	5,556	2,778	2,778	11,112
Provision PT-838	Security camera	5,555	2,778	2,778	11,111
Samsung SNH 1011 N	Webcam	5,555	2,778	2,777	11,110
SimpleHome XCS7-1002-WHT	Security camera	5,555	2,777	2,777	11,109
SimpleHome XCS7-1003-WHT	Security camera	5,555	2,777	2,777	11,109
Total	—	50,000	25,000	25,000	100,000

After constructing the device-level subset, the data were divided into training, validation, and testing subsets using stratified sampling with a 70:15:15:15 ratio. Stratification was applied to preserve the distribution of benign, Mirai, and BASHLITE samples across the three subsets. Accordingly, the final split contained 70,000 samples for training, 15,000 samples for validation, and 15,000 samples for testing.

To avoid data leakage, feature standardization was performed after the train–validation–test split. The Standard Scaler was fitted only on the training subset, and the learned scaling parameters were then applied to the validation and testing subsets. The validation or testing data was not used for training the model due to this process. The corrected preprocessing and sampling procedure therefore supported a fairer and more reliable evaluation of the three models.

This dataset incorporates 115 traffic features with descriptive temporal behavior, packet statistics, and communication statistics. To remove scale effects and stabilize optimization, all features were standardized with the Standard Scaler so that they have a mean value of 0 and a standard deviation of 1. The processed N-BaIoT dataset utilized in this paper is illustrated in **Fig. 4.**

The ANN structure consisted of an input layer with 115 neurons, followed by three hidden layers containing 128, 64, and 32 neurons. The output layer consisted of one neuron with sigmoid activation to predict the probability of botnet attack traffic. A dropout rate of 0.30 was applied to the hidden layers to reduce overfitting. Early stopping with a patience value of 10 was also used to stop training when no further improvement was observed in the validation loss.

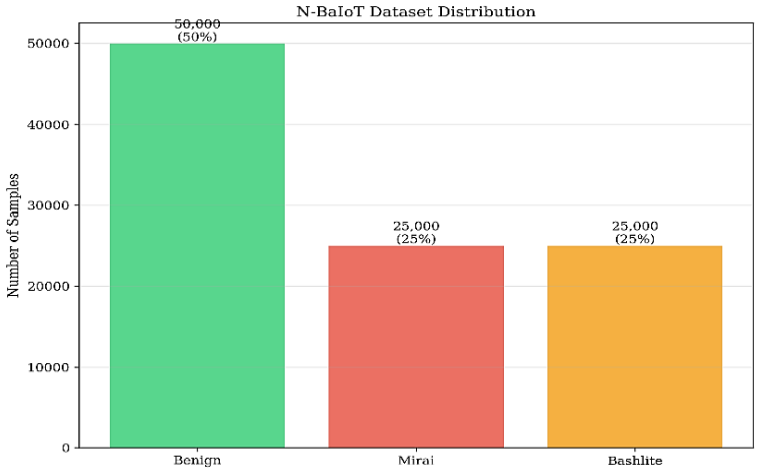

Figure 4. The N-BaIoT preprocessed dataset compatibility with benign, Mirai and Bashlite samples.

3.2 Overall Classification Performance

Table 8 provides the overall classification performances of the three models. Machine learning methods, such as a supervised algorithm, can detect IoT botnets as they scored over 98% accuracy in predictive performance.

Table 8. Comparison of the performance of RF, XGBoost and ANN.

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FAR (%)	AUC	Training Time (s)	Testing Time (ms)
Random Forest	98.45	98.23	98.67	98.45	1.55	0.994	45.3	0.021
XGBoost	99.12	98.98	99.26	99.12	0.88	0.998	52.7	0.018
ANN	98.87	98.65	99.09	98.87	1.13	0.996	120.4	0.035

XGBoost is the approach that has exhibited the best overall performance with an accuracy of 99.12%, a recall of 99.26, F1 score of 99.12 and AUC of 0.998. Moreover, it generated the least False Alarm Rate (0.88%) which is an important property for practical IDSs as an overload of too many false alarms to the network administrators can drain out the limited resources at the edge.

Although the numerical performance margin may not be significant, it has great operational significance. In situations of high-volume IoT traffic, reducing false alarms by even a small amount reduces unnecessary alerts significantly, thus making near-real-time deployment viable. The ANN was also an efficient classifier, showing stronger results than Random Forest most of the time, albeit at a higher cost. The Random Forest technique has the lowest training time and is very competitive in its exercise, though its slightly higher FAR suggests its discrimination capacity is lower concerning borderline traffic.

The applicability of the findings was reinforced by the cross-validation results. The mean cross-validation F1-score for XGBoost was 99.05 (±0.12), Random Forest 98.32 (±0.15) ANN 98.76 (±0.18). It can be observed from the results that the models generalized well due to the stable results across folds. Thus, there is no high variance seen within the folds.

3.2.1 ROC Analysis

As illustrated in **Fig. 5**, the ROC curves confirm that XGBoost outperforms others.

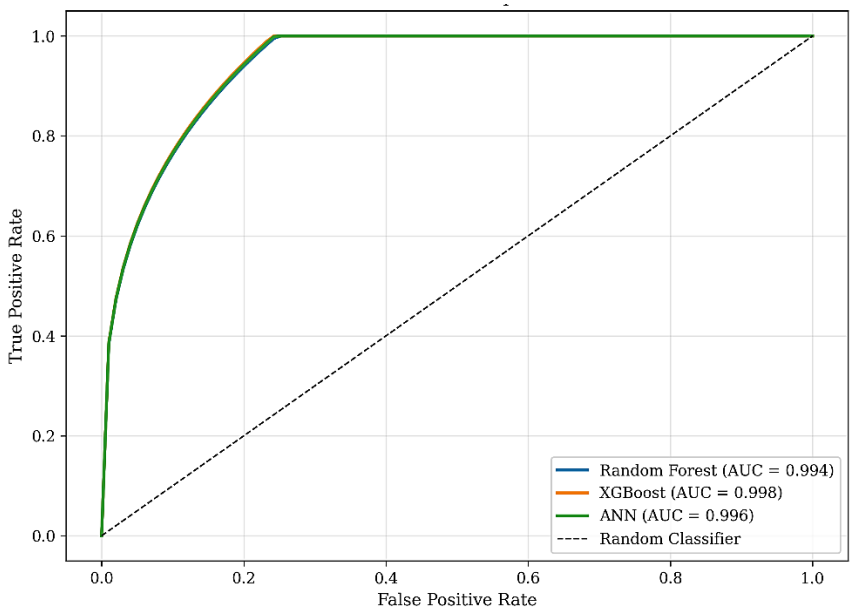

Figure 5. The ROC curves of the models assessed.

According to this, the model that achieved the highest AUC (0.998) was the XGBoost model. Followed by the ANN (0.996) and Random Forest (0.994). As evident from their proximity to the upper left of the ROC space, all the models exhibit remarkable separation between benign and malicious traffic. Still, XGBoost exhibited a superior TPR/FPR trade-off across classification thresholds. Based on the research, the capability to maintain sensitivity without elevations in false alarms is stronger, and this may be useful in adaptive intrusion detection settings where threshold tuning would differ on deployment needs.

3.2.2 Confusion Matrix Analysis

The confusion matrices displayed in **Fig. 6** depict the model behavior associated with each class.

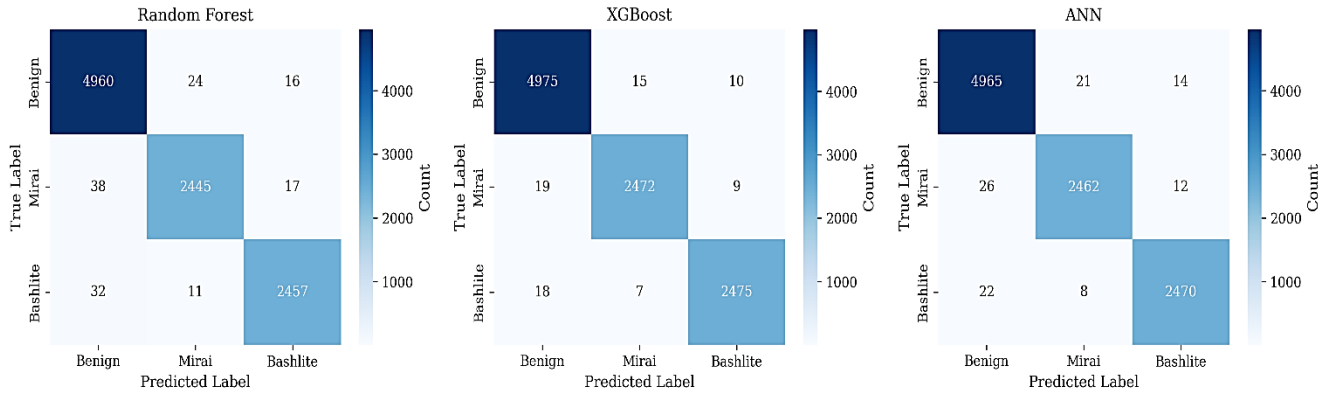

Figure 6. Confusion matrices for (a) RF, (b) XGBoost, and (c) ANN

Each of the three models displayed substantial diagonal dominance, indicating the correct categorization of benign samples, Mirai, and Bashlite traffic. Nonetheless, XGBoost achieved the clearest class separation, with the fewest off-diagonal misclassifications. Compared to other attack scenarios, confusion between benign traffic and Mirai samples is higher for Random Forest. Thus, this explains it is having a higher false alarm rate. ANN outpaced Random Forest in performance, but it is still lower than XGBoost in the ability to discriminate against subtle attacks.

Misclassified samples indicate that most errors occurred early on during attack behavior when patches contained malicious traffic with similar characteristics and normal communication. The aforementioned pattern fits well with the reconnaissance or low-intensity attack phases, where Botnet behavior has not yet developed a strong signature. These observations imply that garnering context in a sequence-wise manner can improve early attack recognition.

3.3 Model-Specific Analysis

3.3.1 Random Forest

The overall accuracy of Random Forest was 98.45%, whereas the precision, recall and F1-score were 98.23%, 98.67% and 98.45%. The model was able to detect benign traffic at a remarkable 99.2% detection rate. Mirai detection achieved 97.8%, while that of Bashlite was 98.3%. As shown in **Fig. 7**, the Random Forest classifier relied mainly on temporal features and packet size features among others.

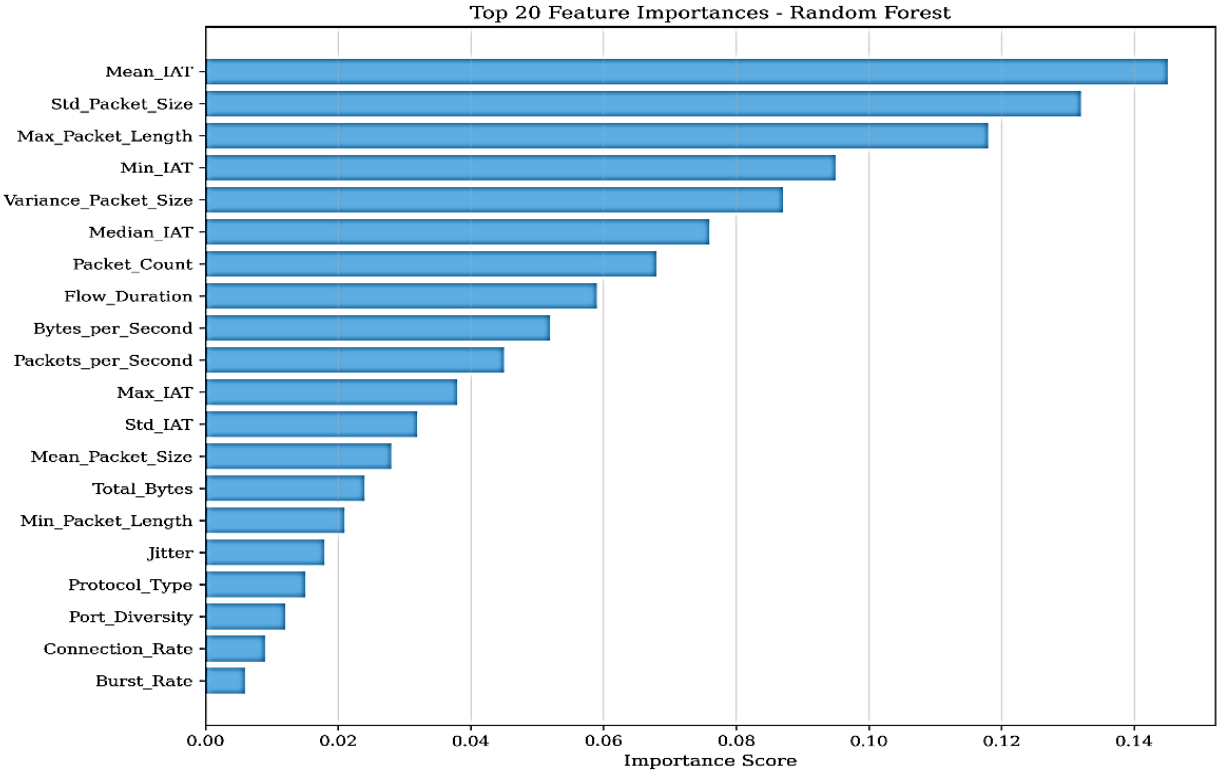

Figure 7. Top 20 feature importance scores for Random Forest

The most important feature was the Mean Inter-Arrival Time (IAT), while packet size dispersion and maximum packet length were also important. The minimum IAT and

variance of the packet size were also strong predictors. The results indicate that attack packet traffic is not only different from benign communication in terms of size but also in terms of timing regularity. And statistical similarity, both of which are captured effectively by ensemble tree methods.

The model utilized a moderately large number of predictors instead of relying heavily upon a small subset of features, as the top ten features accounted for almost 68% of the total contributions. While this behavior may have enhanced its robustness, it may have also contributed to its slightly lower discriminative efficiency compared to XGBoost, which appears to make use of more selective and higher-order interactions. The Random Forest algorithm exhibited the shortest training time of 45.3 seconds, indicating its efficiency for applications requiring frequent retraining or with limited computing resources.

3.3.2 XGBoost

XGBoost excelled in all the major metrics. The F1 score was 99.12%, and FAR was found to be 0.88%. The analysis yielded significant results and the above details. It is likely that the model described here was very accurate and reliable in practice. As per **Table 9,** the outcomes of detection class-wise

Table 9. Class-specific detection accuracy of evaluated models.

Algorithm	Benign (%)	Mirai (%)	Bashlite (%)	Overall (%)
Random Forest	99.2	97.8	98.3	98.45
XGBoost	99.5	98.9	99.0	99.12
ANN	99.3	98.5	98.8	98.87

XGBoost achieved the highest detection rates for benign traffic (99.5%) and Mirai (98.9%). It is a remarkable fact that all classes perform equally well without any discernible bias towards the majority class. In security applications, such consistency is crucial, as models with high overall accuracy could still fail in practice if their performance is concentrated on just one traffic category. The feature importance trend presented in **Fig. 8** demonstrates a greater focus on temporal dynamics than that of Random Forest.

While there was general overlap in the dominant variables and those important to Random Forests, XGBoost assigned even more relative importance to IAT-derived features than Random Forests. This indicates the boosting mechanism took advantage of relatively small timing irregularities associated with Botnet activities. Analysis of the most prominent features revealed a concentrated representation of discriminative traffic characteristics as they accounted for around 72% of total importance.

When it comes to efficiency, XGBoost had a higher training time than Random Forest (52.7 s). However, it has the fastest inference time of 0.018 ms/sample. With a throughput of approximately 55,500 samples/second, the solution is fit for near-real-time deployment at IoT gateways and edge monitoring nodes.

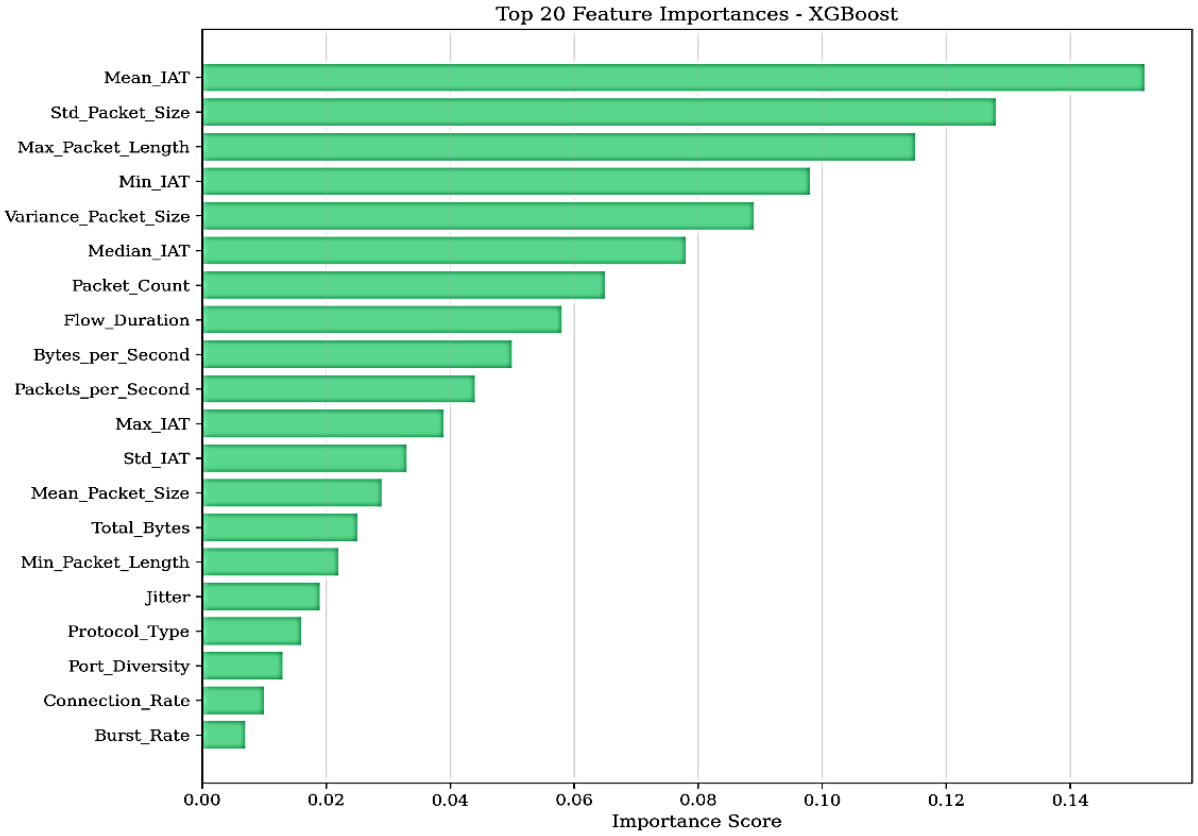

Figure 8. Top 20 feature importance scores for XGBoost

3.3.3 Artificial Neural Network

The accuracy was 98.87%, the precision was 98.65%, the recall was 99.09%, and the F1 measure was 98.87%. Thus, the neural models can also give robust detection capability for IoT Botnet traffic. The learning curves in **Fig. 9** show that this optimization was well stabilized.

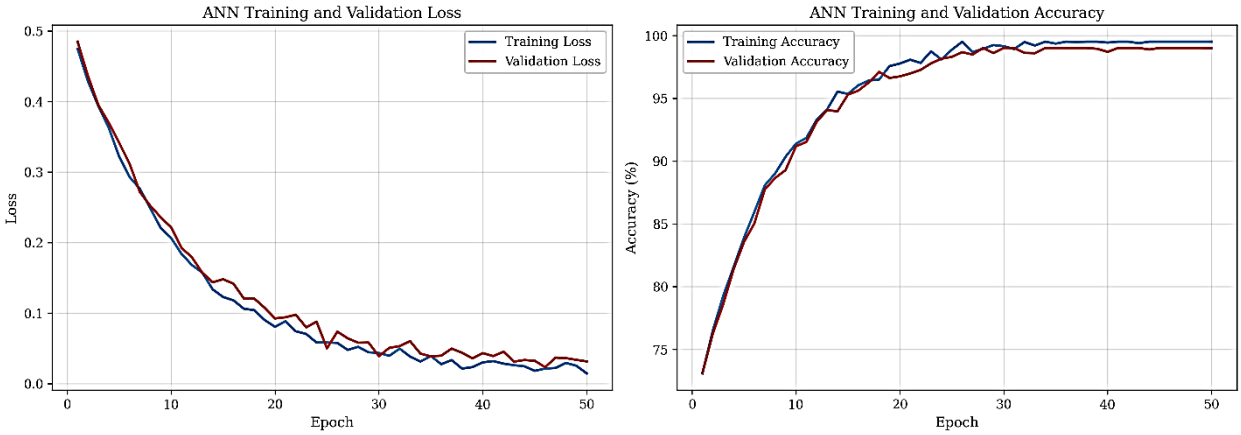

Figure 9. Curves of Training and Validation of ANN Model.

The training and validation performance did not exhibit separation between the two curves during development, making it a smooth process. They did not diverge, which shows the model generalized well. After around 30 epochs, the validation loss stabilized, suggesting that early stopping was effective in preventing overfitting and saving on training resources.

At the class level, ANN achieved an accuracy of 99.3% for benign traffic, 98.5% for Mirai and 98.8% for Bashlite. These values indicate competitive performance, with ANN consistently lagging slightly behind XGBoost across all categories. Although the predictive power of the ANN was strong, ANN itself faced computational issues. It required over 120.4 s for training and 0.035 ms per sample for inference. Although still suitable for a variety of experimental and centralized settings, this renders ANN less appealing for highly constrained near-real-time deployment scenarios.

3.4 Class-Level Detection Behavior

Fig. 10 compares the class-specific detection rates achieved by the three models.

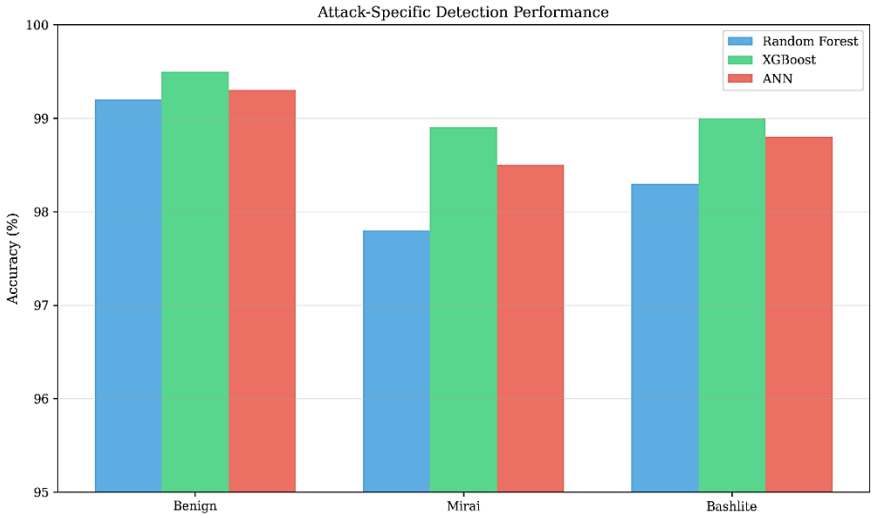

Figure 10. Comparative class-wise detection performance

The classifiers achieved a 99% accuracy on benign traffic. Having an accurate benign classification is useful to the extent that it will reduce the false positive alerts in operation. XGBoost had the best benign detection rate of 99.5% followed by ANN and Random Forest at 99.3% and 99.2% respectively.

All models found Mirai detection slightly harder. The highest Mirai detection accuracy was achieved by XGBoost at 98.9% followed by ANN at 98.5 and Random Forest at 97.8%. The reason for this relative decline might be that early Mirai traffic may partially resemble legitimate scanning or discovery behavior conducted by IoT devices.

Results with all models are better than for Mirai. For the Bashlite dataset, XGBoost had the best performance with an accuracy of 99.0%. An ANN produced similar results with a 98.8% accuracy. Random Forest was a little lower with an accuracy of 98.3%. As such, Bashlite traffic patterns are, on average, more distinguishable in the considered feature space.

The analysis of the errors revealed two trends. To begin with, false positives tended to occur when there has been a legitimate but unusual traffic burst. This could be due to a device update or a transient spike in communication between electronic devices. The second point to note is that the false negatives mostly concentrated on low-intensity and early-stage attack behavior with the malicious signature not being pronounced enough. It is noteworthy that the confusion between Mirai and Bashlite is minimal, indicating that the selected features were effective in distinguishing the two attack families.

3.5 Computational Efficiency

In IoT Security Systems, simply achieving high detection accuracy is not enough. The chosen model should also comply with resources and near-real-time specifications. All algorithms tested are more than easy and cheap to compute.

Table 10. Efficiency in computation and model size.

Algorithm	Training Time (s)	Testing Time per Sample (ms)	Throughput (samples/s)	Model Size (MB)
Random Forest	45.3	0.021	~47,600	12.3
XGBoost	52.7	0.018	~55,500	8.7
ANN	120.4	0.035	~28,500	15.6

Table 10 shows that Random Forest has the lowest training time, which may be appropriate when models need to be retrained often as traffic patterns change. In contrast, XGBoost showed the best trade-off between learning costs and runtime. The system had a slightly longer training period, but it achieved the fastest inference speed with the least memory. Due to the decision latency and model compactness, such properties are crucial for edge-assisted IoT security architectures and deployments.
ANN was the computationally expensive (in terms of training and inference) model. While it has good predictive performance, it uses too many resources and is not ideal for low-power or low-latency contexts. **Fig. 11** shows that different Models have different Testing and training times.

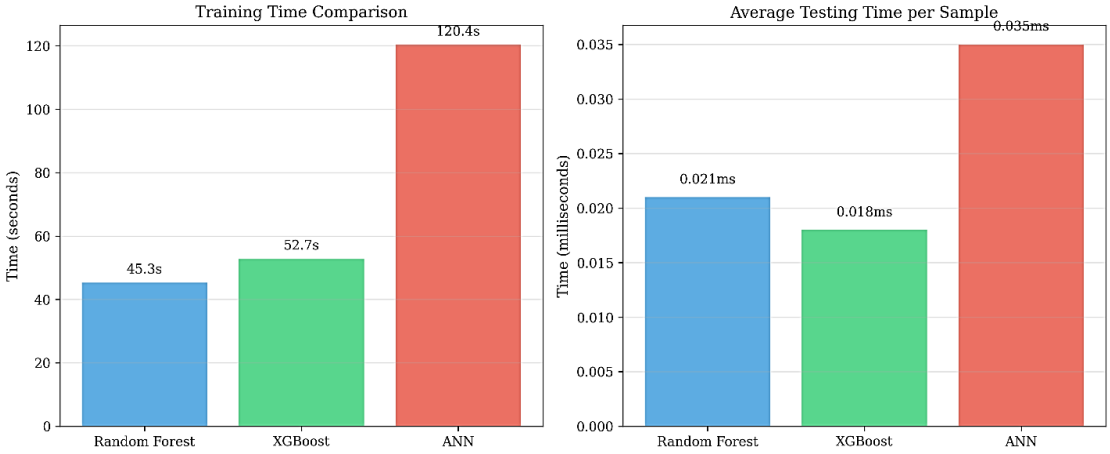

Figure 11. Comparison of models regarding time for training and inference.

The results indicate that XGBoost is the most feasible model when predictive capability and operational efficiency are considered together.

3.6 Integrated Interpretation of Results

Percentage-wise comparison of the main performance indicators is depicted by **Fig. 12.**

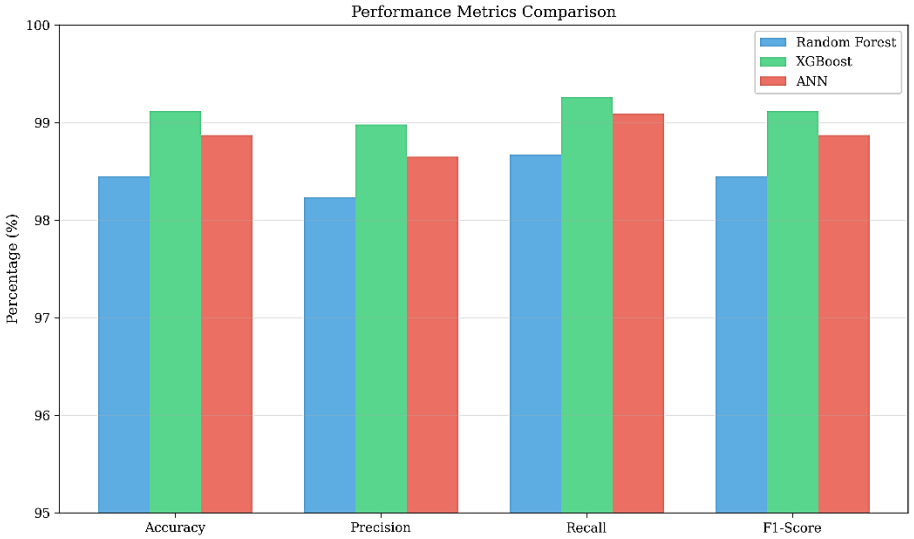

Figure 12. Comparative visualization of accuracy, recall, F1-score, FAR, and AUC

Three patterns emerge based on the findings. According to the initial analysis, it was found that the three models are able to carry out the detection of IoT traffic Botnet attacks above normal. The feature set revealed an extractive nature and discriminative power. In the second place, XGBoost achieves a good balance across sensitivity, precision and false alarm control and is the most reliable candidate. The ensuing results of the study demonstrate that temporal traffic characteristics, i.e. inter-arrival time statistics, became the most significant predictor in each case. Thus, the timing behavior can play an important role in discriminating Botnet traffic from the normal traffic of smart devices.

While the models perhaps differ by a small absolute value, the difference becomes large in deployed conditions. A cut in FAR from 1.55% to 0.88% will bring a quantum improvement in the quality and efficiency of alerts. In an overly crowded IoT setting, anything around a false alarm rate of 1% can create an extremely large absolute number of false alarms so this is important.

3.7 Summary of Key Findings

Results of the research show that the implementation of machine learning for detection of IoT Botnet attacks is possible. The models which were monitored and had accuracy above 98 percent were fitted using supervised learning. XGBoost, nevertheless, proved to be the most effective system, yielding the best classification performance overall, the smallest false alarm rate, the highest AUC, the fastest inference speed, and the smallest model size.

Random Forest is computationally efficient and provides strong performance, particularly where training speed is the main concern. ANN also showed competitive classification capability. However, its higher computative requirement limited ANN overall deployment advantage over tree-based methods.

According to the class-wise evaluation, Mirai traffic was slightly more difficult to detect than BASHLITE traffic. Analysis of feature importance reaffirmed that the temporal traffic dynamics are the most differentiating feature of malicious behavior. In general, the results support the choice of XGBoost as the best model for near-real-time IoT Botnet detection in resource-aware security environments. The present findings affirm the high effectiveness of machine learning to detect IoT botnet from the structured traffic features. The accuracy of all the models evaluated was more than 98%. XGBoost was found to give the best results

among all of them for accuracy, recall, F1 score, and AUC, but also had the lowest false alarm and fastest inference. This suggests that XGBoost not only provides a powerful predictive performance but also offers some suitability to be deployed in a near-real-time IoT security environment. The findings echo recent research highlighting the significance of resource-efficient ensemble-based and lightweight detection methods in network-restricted IoT settings **(Hossain and Islam, 2025; Khan et al., 2025).**

XGBoost has performance superiority. This is due to its boosting approach that puts focus on hard samples. It also improves decision boundaries in comparison to the standard average ensemble. In IoT botnet detection, early malicious behavior may look partially like benign communication. Further analysis of feature importance showed that temporal traffic features are the most relevant variable across all models. More specifically, inter-arrival-time-related variables were the most important predictors across all models. According to this observation, timing behavior is an important discriminative signal that can distinguish between bot traffic and human activity within the IoT. According to interpretation, feature optimization and efficient feature representation are essential as these impact the performance of IoT intrusion detection, according to recent studies **(Liu and Du, 2023; Baker and Samarneh, 2024; Li et al., 2024).**

Even though MLP performed competitively, it did not outperform XGBoost. The tabular structure of the dataset used in the study supports this conclusion. For such data, tree-based methodologies usually outperform neural models, as they can facilitate nonlinear interactions among engineered features without the higher cost of deep learning. The performance of Random Forest too was good, and it remained attractive in terms of training speed, but it was slightly less effective in minimizing false alarms and separating difficult classes as compared to XGBoost. Similar observations can be made in settings dealing with N-BaIoT and other IoT botnet detection benchmarks. Particularly, classical ensemble methods are often still very competitive as deep architectures grow popular **(Kim et al., 2020; Wardana et al., 2024).**

At the classifier level, all models performed best on benign traffic. Also, Mirai is slightly harder to detect than Bashlite. Some Mirai behaviors, thus, correspond more closely to legitimate traffic patterns in low intensity or early phases. Error analysis revealed that most cases were false positives. According to the findings, the true positive alerts were related to false but benign traffic burst incidents, whereas the false negatives were mainly in subtle stages of attacks that were insufficiently distinctive, statistically developed or signatured. The rich modeling of time and the recourse to broader contexts may offer clues to future improvement. A recent study on hybrid and sequence-aware detection models shows that temporal learning may improve sensitivity to emerging or low-intensity attack behavior **(Hashim et al., 2024; Kamal and Mashaly, 2025).**

In contrast to a lot of existing literature, the present study provides a more even-handed assessment, integrating overall metrics and class-specific analyses, along with feature importance interpretation and computational efficiency in a single experimental framework. Instead of being solely concerned with accuracy, the study shows that IoT intrusion detection should consider the false alarm rate, inference speed, and deployment constraints. XGBoost was the most favored solution under this regard as it achieved strong detection performance at low runtime cost, and the model size is compact. A more deployment-oriented viewpoint has been increasingly presented in recent IoT security research activities, notably in lightweight architecture studies, efficient inference and explainable decision support **(Khan et al., 2025; Saied and Guirguis, 2025; Kalakoti et al., 2024).**

Notwithstanding those strengths, the study has limitations. The experiments were run on a single dataset only, the N-BaIoT, and only on Mirai and Bashlite. Consequently, the findings cannot necessarily be generalized to other botnet families or real-world traffic. Additionally, the model relies on engineered tabular features instead of the sequential traffic data which is likely to leave out potentially useful temporal patterns that more sophisticated sequence-aware models may be able to capture. In the end, the evaluation was offline not taking into account concept drift, adversarial tweaking, or maintenance of the model over time.
Research should therefore be conducted to validate the models in more heterogeneous IoT datasets, sequence based or hybrid detection architectures should be studied, and online learning methods should be studied to maintain performance to change traffic and attacks. Explanations regarding predictive model decisions can be trusted by the analysts to back operational decision-making **(Saied and Guirguis, 2025; Thein et al., 2024).**
According to researchers, machine learning can be used to detect IoT botnet effectively. Further, among the models studied, the XGBoost model had the best performance concerning consumption of resources and detection accuracy.

4. CONCLUSIONS

In this analysis, Random Forest, XGBoost, and Artificial Neural Network models were tested for their ability to effectively detect IoT botnets by employing the N-BaIoT dataset. The study revealed that all three models could distinguish benign traffic from Mirai and BASHLITE attacks with more than 98% classification performance.
Out of all the models that were evaluated, XGBoost produced the best overall performance with 99.12% accuracy, 98.98% precision, 99.26% recall, 99.12% F1-score, 0.998 AUC and fastest inference time of 0.018 ms/sample. As a result, the model that best achieved detection accuracy and the computational time was XGBoost and, thus, it is the most suitable one for near-real-time IoT botnet detection at gateway or edge-monitoring levels.
The Random Forest method yielded satisfactory results as well producing 98.45% accuracy and F1-score and the shortest training time. The ANN model attained competitive results of 98.87% accuracy and 0.996 AUC but had higher training and inference costs. Analysis of Feature Importance show that temporal traffic features, in particular inter-arrival-time-related features, were among the most discriminative indicators.
A selected subset of the N-BaIoT dataset and two botnet families were the scope of the study were Mirai and BASHLITE. In future research, it is desired that the models be validated using larger and more recent IoT datasets, more botnet families, and in real deployments. Future enhancements may include strategies such as feature selection, sequence-aware models, hybrid learning techniques and online adaptation.

Declaration of Competing Interest

The author declares that he does not have competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

Abbasi, M., Farokhnia, A., Bahreinimotlagh, M. and Roozbahani, R., 2021. A hybrid of Random Forest and Deep Auto-Encoder with support vector regression methods for accuracy improvement and uncertainty reduction of long-term streamflow prediction. *Journal of Hydrology*, *597*, P. 125717. https://doi.org/10.1016/j.jhydrol.2020.125717

Alharbi, A., Alosaimi, W., Alyami, H. and Rauf, H.T., 2021. Botnet attack detection using local global best bat algorithm for industrial Internet of Things. *Electronics*, 10(11), P. 1341. https://doi.org/10.3390/electronics10111341

Alqahtani, A., Alsulami, A.A., Alqahtani, N., Alturki, B. and Alghamdi, B.M., 2024. A comprehensive security framework for asymmetrical IoT network environments to monitor and classify cyberattack via machine learning. *Symmetry*, 16(9), P. 1121. https://doi.org/10.3390/sym16091121 .

Baker, Q.B. and Samarneh, A., 2024. Feature selection for IoT botnet detection using equilibrium and Battle Royale Optimization. *Computers & Security*, 147, P. 104060. https://doi.org/10.1016/j.cose.2024.104060 .

Cao, Y., Wang, Z., Ding, H., Zhang, J. and Li, B., 2023. IoT botnet attacks detection and classification based on ensemble learning. *In International Symposium on Artificial Intelligence and Robotics,* pp. 45-55. Singapore: Springer Nature Singapore. https://doi.org/10.1007/978-981-99-9109-9_5 .

Catillo, M. and Pecchia, A., 2022. Botnet detection in the Internet of Things through all-in-one deep autoencoding. *In: Proceedings of the 17th International Conference on Availability, Reliability and Security (ARES '22)*. https://doi.org/10.1145/3538969.3544460 .

Catillo, M., Pecchia, A. and Villano, U., 2023. A deep learning method for lightweight and cross-device IoT botnet detection. *Applied Sciences*, 13(2), P. 837. https://doi.org/10.3390/app13020837 .

Cunha, A.A., Borges, J.B. and Loureiro, A., 2022. Classification of botnet attacks in IoT using a convolutional neural network. *In: MSWiM '22: Proceedings of the 25th International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. https://doi.org/10.1145/3551661.3561374 .

Dietterich, T.G., 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7), pp. 1895–1923. https://doi.org/10.1162/089976698300017197 .

Fawcett, T., 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), pp. 861–874. https://doi.org/10.1016/j.patrec.2005.10.010 .

Garg, U., Kumar, S. and Mahanti, A., 2024. IMTIBOT: An intelligent mitigation technique for IoT botnets. *Future Internet*, 16(6), P. 212. https://doi.org/10.3390/fi16060212 .

Gelenbe, E. and Nakip, M., 2022. Traffic based sequential learning during botnet attacks to identify compromised IoT devices. *IEEE Access*, 10, pp. 126536–126549. https://doi.org/10.1109/ACCESS.2022.3226700 .

Gelgi, M., Guan, Y., Arunachala, S., Samba Siva Rao, M. and Dragoni, N., 2024. Systematic literature review of IoT botnet DDoS attacks and evaluation of detection techniques. *Sensors*, 24(11), P. 3571. https://doi.org/10.3390/s24113571 .

Hashim, W.A., Ali, S.S.M., Al-Khawaldeh, A. and Al-Shammri, F.K., 2024, December. Botnet Detection Using Hybrid Methods. In *2024 International Jordanian Cybersecurity Conference (IJCC)*, pp. 21-27. IEEE. https://doi.org/10.1109/IJCC64742.2024.10847282

Hossain, M.A. and Islam, M.S., 2025. A novel feature selection-driven ensemble learning approach for accurate botnet attack detection. *Alexandria Engineering Journal*, 118, pp. 261–277. https://doi.org/10.1016/j.aej.2025.01.042 .

Kalakoti, R., Bahsi, H. and Nomm, S., 2024. Improving IoT security with explainable AI: Quantitative evaluation of explainability for IoT botnet detection. *IEEE Internet of Things Journal*, 11(10), pp. 18237–18254. https://doi.org/10.1109/JIOT.2024.3360626 .

Kamal, H. and Mashaly, M., 2025. Robust intrusion detection system using an improved hybrid deep learning model for binary and multi-class classification in IoT networks. *Technologies*, 13(3), P. 102. https://doi.org/10.3390/technologies13030102 .

Kaufman, S., Rosset, S., Perlich, C. and Stitelman, O., 2012. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data*, 6(4), Article 15. https://doi.org/10.1145/2382577.2382579 .

Khan, M.R.A., Barnawi, A.Y., Munir, A., Alsalman, Z. and Sanunga, D.M.S., 2025. Lightweight quantized XGBoost for botnet detection in resource-constrained IoT networks. *IoT*, 6(4), P. 70. https://doi.org/10.3390/iot6040070 .

Kim, J., Shim, M., Hong, S., Shin, Y. and Choi, E., 2020. Intelligent detection of IoT botnets using machine learning and deep learning. *Applied Sciences*, 10(19), P. 7009. https://doi.org/10.3390/app10197009 .

Kumar, A., Shridhar, M., Swaminathan, S. and Lim, T.J., 2022. Machine learning-based early detection of IoT botnets using network-edge traffic. *Computers & Security*, 117, P. 102693. https://doi.org/10.1016/j.cose.2022.102693 .

Lefoane, M., Ghafir, I., Kabir, S. and Awan, I.-U., 2025. Internet of Things botnets: A survey on artificial intelligence-based detection techniques. *Journal of Network and Computer Applications*, 237, P. 104110. https://doi.org/10.1016/j.jnca.2025.104110 .

Li, J., Othman, M.S., Chen, H. and Yusuf, L.M., 2024. Optimizing IoT intrusion detection system: Feature selection versus feature extraction in machine learning. *Journal of Big Data*, 11, P. 36. https://doi.org/10.1186/s40537-024-00892-y .

Liu, X. and Du, Y., 2023. Towards effective feature selection for IoT botnet attack detection using a genetic algorithm. *Electronics*, 12(5), P. 1260. https://doi.org/10.3390/electronics12051260 .

Malik, K., Rehman, F., Maqsood, T., Mustafa, S., Khalid, O. and Akhunzada, A., 2022. Lightweight Internet of Things botnet detection using one-class classification. *Sensors*, 22(10), P. 3646. https://doi.org/10.3390/s22103646 .

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A. and Elovici, Y., 2018. N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17(3), pp. 12–22. https://doi.org/10.1109/MPRV.2018.03367731 .

Nazir, A., He, J., Zhu, N., Wajahat, A., Ma, X., Ullah, F., Qureshi, S. and Pathan, M.S., 2023. Advancing IoT security: A systematic review of machine learning approaches for the detection of IoT botnets. *Journal of King Saud University - Computer and Information Sciences*, 35(9), P. 101820. https://doi.org/10.1016/j.jksuci.2023.101820 .

Powers, D.M.W., 2011. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1), pp. 37–63.

Saied, M. and Guirguis, S., 2025. Explainable artificial intelligence for botnet detection in internet of things. *Scientific Reports*, 15, P. 7632. https://doi.org/10.1038/s41598-025-90420-6 .

Serhane, A., Ibrahimi, K., Hamzaoui, E.-M., Jouhari, M. and others, 2025. IoT intrusion detection using machine learning classifiers and PCA dimensionality reduction for N-BaIoT dataset. *In: ICC 2025 - IEEE International Conference on Communications*. https://doi.org/10.1109/ICC52391.2025.11161305 .

Sokolova, M. and Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), pp. 427–437. https://doi.org/10.1016/j.ipm.2009.03.002 .

Thein, T.T., Shiraishi, Y. and Morii, M., 2024. Personalized federated learning-based intrusion detection system: Poisoning attack and defense. *Future Generation Computer Systems*, 153, pp. 182–192. https://doi.org/10.1016/j.future.2023.10.005

Wang, Z., Li, J., Yang, S., Luo, X., Li, D. and Mahmoodi, S., 2024. A lightweight IoT intrusion detection model based on improved BERT-of-Theseus. *Expert Systems with Applications*, 238, P. 122045. https://doi.org/10.1016/j.eswa.2023.122045 .

Wardana, A.A., Kołaczek, G., Warzyński, A. and Sukarno, P., 2024. Ensemble averaging deep neural network for botnet detection in heterogeneous *Internet of Things devices*. Scientific Reports, 14, P. 3878. https://doi.org/10.1038/s41598-024-54438-6

Wu, Y., He, X. and Chen, X., 2022. IoT-botnet traffic detection based on deep forest. *In: 2022 IEEE 22nd International Conference on Communication Technology (ICCT)*, pp. 1388–1393. https://doi.org/10.1109/ICCT56141.2022.10072774 .

الكشف عن هجمات البوت نت في شبكات إنترنت الأشياء بالاعتماد على تعلّم الآلة: دراسة تجريبية مقارنة

علي محمد نوري تراب

قسم الحاسوب والاتصالات، كلية الهندسة، الجامعة الإسلامية في لبنان، لبنان.

الخلاصة

تتميّز الأجهزة المتصلة ضمن بيئات إنترنت الأشياء بمحدودية مواردها الأمنية والحاسوبية، الأمر الذي جعلها أكثر عرضةً للبرمجيات الخبيثة وأسهم في تصاعد استهداف شبكات إنترنت الأشياء بهجمات البوت نت المتقدمة. تهدف هذه الدراسة إلى إجراء تقييم تجريبي مقارن لعدد من نماذج التعلّم الآلي المستخدمة في الكشف عن هجمات البوت نت اعتماداً على مجموعة بياناتN-BaIoT ، التي تتضمن حركة مرور حقيقية لأجهزة إنترنت الأشياء في الحالات الطبيعية، إضافةً إلى حركة مرور خبيثة ناتجة عن عائلتي Mirai و .BASHLITEتقارن الدراسة بين ثلاثة نماذج تمثيلية، وهي Random Forest :بوصفه نموذجاً للتعلّم الآلي التقليدي، و XGBoostبوصفه نموذجاً متقدماً لتعلّم الحشد، والشبكة العصبية الاصطناعية ذات التغذية الأمامية الكاملة بوصفها نموذجاً للتعلّم العميق. جرى تقييم النماذج باستخدام الدقة، والدقة التنبؤية، والاستدعاء، وقيمةF1-score، ومعدل الإنذار الكاذب، وقيمةAUC ، وزمن التدريب، وزمن الاستدلال. أظهرت النتائج أن خوارزمية XGBoost حققت أفضل أداء إجمالي، إذ بلغت الدقة 99.12%، والدقة التنبؤية 98.98%، والاستدعاء 99.26%، وقيمة F1-score مقدارها 99.12%، ومعدل الإنذار الكاذب 0.88%، وقيمة AUC مقدارها 0.998. كما سجلت أقل زمن استدلال بلغ 0.018 مللي ثانية لكل عينة، مقارنةً بـ 0.021 مللي ثانية لخوارزمية Random Forest و0.035 مللي ثانية لنموذج .ANN وتشير هذه النتائج إلى أن XGBoost يحقق أفضل توازن بين دقة الكشف والكفاءة الحاسوبية، مما يجعله نموذجاً مناسباً للكشف شبه الفوري عن هجمات البوت نت في بوابات إنترنت الأشياء أو بيئات المراقبة الطرفية.

الكلمات المفتاحية: الكشف عن شبكات الروبوتات، التعلم الآلي الكلاسيكي، التعلم الجماعي، التعلم العميق، الغابة العشوائية، XGBoost، الأمن السيبراني.