telecom

MDPI

Article

XGB-RF: A Hybrid Machine Learning Approach for IoT Intrusion Detection

Jabed Al Faysal [1], Sk Tahmid Mostafa [2], Jannatul Sultana Tamanna [2], Khondoker Mirazul Mumenin [2], Md. Mashrur Arifin [2], Md. Abdul Awal [2,*], Atanu Shome [1] and Sheikh Shanawaz Mostafa [3,*]

[1] Computer Science and Engineering Discipline (CSE), Khulna University (KU), Khulna 9208, Bangladesh; faysal@ku.ac.bd (J.A.F.); atanu@ku.ac.bd (A.S.)
[2] Electronics and Communication Engineering (ECE) Discipline, Khulna University (KU), Khulna 9208, Bangladesh; sktahmidmostafa@gmail.com (S.T.M.); jannatutamanna1.5@gmail.com (J.S.T.); k.mirazulmumenin@gmail.com (K.M.M.); arifinmash@gmail.com (M.M.A.)
[3] ITI—Interactive Technologies Institute, LARSyS, Laboratory of Robotics and Systems in Engineering and Science, M-ITI, ARDITI, 9000 Funchal, Portugal
* Correspondence: m.awal@ece.ku.ac.bd (M.A.A.); sheikh.mostafa@iti.arditi.pt (S.S.M.); Tel.: +880-178-619-6913 (M.A.A.)

Abstract: In the past few years, Internet of Things (IoT) devices have evolved faster and the use of these devices is exceedingly increasing to make our daily activities easier than ever. However, numerous security flaws persist on IoT devices due to the fact that the majority of them lack the memory and computing resources necessary for adequate security operations. As a result, IoT devices are affected by a variety of attacks. A single attack on network systems or devices can lead to significant damages in data security and privacy. However, machine-learning techniques can be applied to detect IoT attacks. In this paper, a hybrid machine learning scheme called XGB-RF is proposed for detecting intrusion attacks. The proposed hybrid method was applied to the N-BaIoT dataset containing hazardous botnet attacks. Random forest (RF) was used for the feature selection and eXtreme Gradient Boosting (XGB) classifier was used to detect different types of attacks on IoT environments. The performance of the proposed XGB-RF scheme is evaluated based on several evaluation metrics and demonstrates that the model successfully detects 99.94% of the attacks. After comparing it with state-of-the-art algorithms, our proposed model has achieved better performance for every metric. As the proposed scheme is capable of detecting botnet attacks effectively, it can significantly contribute to reducing the security concerns associated with IoT systems.

Keywords: IoT security; botnet detection; random forest; XGB; feature selection; Mirai

Citation: Faysal, J.A.; Mostafa, S.T.; Tamanna, J.S.; Mumenin, K.M.; Arifin, M.M.; Awal, M.A.; Shome A.; Mostafa, S.S. XGB-RF: A Hybrid Machine Learning Approach for IoT Intrusion Detection. *Telecom* **2022**, *3*, 52–69. https://doi.org/10.3390/telecom3010003

Academic Editor: Michał Aibin

Received: 26 November 2021
Accepted: 28 December 2021
Published: 4 January 2022

1. Introduction

The Fourth Industrial Revolution will be fueled by cutting-edge and innovative technologies where the Internet of Things (IoT) devices will play integral roles [1]. Countries prepared to accept these changes will undoubtedly have a better chance of success in the near future. IoT is one of the fastest expanding fields in the history of technology, with around 50 billion devices in use by the end of 2020 [2]. As a result, the devices' security threats are increasing on a large scale. Because IoT devices lack fundamental security protocols, they have become tempting targets for attackers. According to recent estimates, IoT devices are subjected to an average of 5200 attacks per month [3]. Only in the first half of 2019, attacks against IoT devices tripled compared with the previous year [4]. According to Checkpoint's study, 71% of security experts have observed an increase in security risks in IoT networks after the prevalence of COVID-19 [5]. Recently, the Internet of Medical Things (IoMT) [6,7], the Long Range (LoRa) IoT networks [8], Blockchain-based supply chain systems [9,10] and Smart industries [11] have also been targeted by a huge number of attackers. Many of these attacks are extremely hazardous for the devices. Mirai and BASHLITE, found in 2016 and 2015, respectively, are two common IoT botnets that

infected numerous IoT devices over the years [12]. Among them, the Mirai Distributed Denial-of-Service (DDoS) attack was the third most frequent IoT threat in 2018. Again, a total of 15.4 million DDoS attacks are expected worldwide by 2023 [13]. So, with the rise in cybercrime, detecting these attacks has become a crucial field of research [14]. There should be an effective framework assuring the security of IoT devices as these issues are highly alarming. Nonetheless, the use of regular security techniques for detecting botnet attacks would not be sufficient to ensure the safety of the systems.

Many researchers have been working on developing intrusion detection systems in recent years, as the attacks increasingly target IoT devices. They mainly focus on two types of botnet attacks in the IoT environment, namely host-based [15], and network-based [16]. However, some of the common machine learning approaches for intrusion detection include support vector machine (SVM), K-nearest neighbor (KNN), neural networks, and Naïve Bayes [17–19]. Shafiq et al. [20] presented a methodology for selecting the efficient machine learning algorithm in IoT devices to detect malicious attacks. They claimed Naïve Bayes as the efficient algorithm according to their experimental analysis. Soe et al. [21] performed a feature selection approach on the N-BaIoT dataset to detect IoT attacks. They obtained nearly 99% accuracy using three distinct machine learning algorithms, including Naïve Bayes, ANN, and J48 decision tree. Diro et al. [22] introduced an intrusion detection method for IoT devices that uses deep learning as a primary tool for detection and achieved significant improvement. Ahmad et al. [23] evaluated the NSL-KDD dataset for intrusion detection using random forest (RF), SVM, and extreme learning machines (ELM) algorithms. Their findings indicated that ELM outperformed SVM when dealing with large data. In another study, a network-based intrusion detection proposed by Deng et al. [24] achieved a detection rate of 96.8% when utilizing K-means clustering after manually selecting 8 to 16 features. Mirsky et al. [25] proposed an unsupervised model which is autonomous. This model is able to observe security issues in network environments where an ensemble autoencoder algorithm is used. This classification model, which operates on low computational resources, is primarily composed of two steps: offline training and online testing on IoT devices. In addition, Radford et al. [26] proposed an unsupervised learning where Long-Short Term Memory (LSTM) cell Recurrent Neural Networks (RNN) is used for network traffic anomaly detection by utilizing a public dataset (https://www.unb.ca/cic/datasets/ids-2017.html (accessed 30 September 2021)). However, no research has provided an in-depth analysis of machine learnings' applicability in the field of host-based intrusion detection for IoT environments [27].

From the literature, it is clear that machine learning-based techniques, being used widely in the IoT security domain, are proven to be effective for developing workable models for detecting IoT threats. As most of the attacks happen in real-time in the IoT environment, therefore a fast attack detection algorithm is required while maintaining higher accuracy. This could be possible by using fewer features that reduce the system complexity and consequently, it should execute faster. This could be very helpful in real-time attack detection. To solve this issue, we propose a machine learning-based approach for intrusion detection in IoT systems where the random forest (RF) algorithm is used to select inevitable features from the N-BaIoT dataset to boost detection accuracy. Then, the eXtreme Gradient Boosting algorithm (XGB) is utilized to characterize and detect malicious attacks. This novelty is very essential for IoT-based intrusion detection as here we have to deal with a big dataset where both time and accuracy are very crucial. However, when RFE and RFECV are used, attributes are added recursively in a one-by-one fashion and check whether that attribute improves the performance or not. If it doesn't improve the classification performance, then that attribute is discarded. In addition, this process continues until all the attributes are checked. On the other hand, RF ranks the features according to information gain rather than checking the feature importance of each feature one-by-one basis. After that, the proposed system is compared with some other state-of-the-art algorithms applied on the same N-BaIoT dataset, where our system achieves much better performance in every case. The key contributions of this paper are as follows:

- A hybrid machine learning algorithm has been proposed, for the first time, named XGB-RF where the prominent attributes are selected using RF algorithm and then classified using XGB algorithm.
- The proposed algorithm has also been compared with other state-of-the-art machine learning algorithms.

2. Materials and Methods

The hybrid method proposed for detecting in N-BaIoT multi-class attack is called XGB-RF classification. Firstly, the RF feature selection algorithm is used for selecting feasible features. Then XGB classification algorithm is to identify each type of attack that occurs in the IoT network. For this, the proposed method is named XGB-RF. The multiplicity of the N-BaIoT dataset is reduced from 115 to 40 features that relate to the same window using our proposed strategy. The following diagram (Figure 1) represents our proposed system architecture.

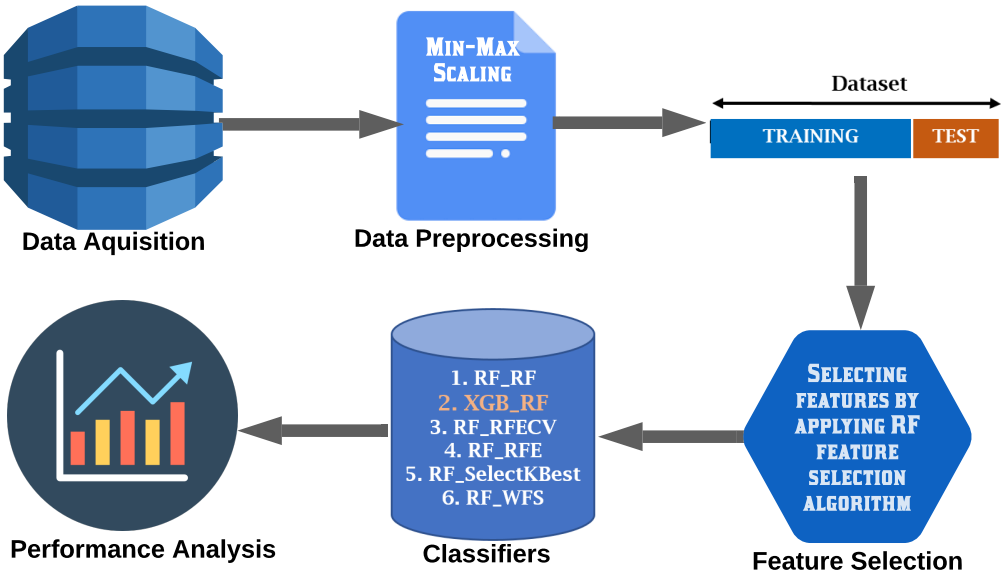


Figure 1. Workflow diagram of the proposed system.

2.1. Data Acquisition

The research has considered the N-BaIoT dataset [28] to detect IoT botnet attacks, one of the most recent datasets published in 2020 and publicly available (https://www.kaggle.com/mkashifn/nbaiot-dataset (accessed 5 June 2021)). The dataset is comprised of nine IoT sensor traffic captured in a local network using Wireshark in the central switch. It has 115 statistically designed attributes derived from the pcap files. Over five separate time windows, seven statistical measures (mean, variance, count, magnitude, radius, covariance, and correlation coefficient) were computed. Since it includes a time window, this dataset is suited for a stateful Intrusion Detection System (IDS). It is a predetermined timeframe during which data are extracted from traffic within the same period. However, three or more statistical measures were computed for each of these four features, yielding a total of twenty-three features.

A total of 229,829 samples were used; among them, benign and malicious attack samples are 13,113 and 216,716, respectively. The malicious attack consists of 27,188, 9502, 23,361, 15,148, 26,210, 21,205, 24,250, 21,995, 23,755, 24,102 samples of Mirai.ack, Mirai.scan, Mirai.syn, Mirai.udp, Mirai.udpplain, Gafgyt.combo, Gafgyt.junk, Gafgyt.scan, Gafgyt.tcp, Gafgyt.udp, respectively. In this study, a multi-class (i.e., 11 classes) dataset was used. A description of these classes is provided as below:

- **Benign (Class_1)**: The benign class is defined as the traffic which does not carry any attack or malicious activities.
- **Mirai ACK & Mirai SYN (Class_2 & Class_4)**: In order to prevent legitimate clients from connecting to a server, Mirai uses SYN flooding. As soon as the two parties complete the three-way handshake: the client sends SYN, receives SYN, the server sends ACK+1, the client sends ACK+1 (which the server receives and processes), and the information exchange may begin. Mirai (the threat actor) uses an illegal IP address to send SYN queries to all open ports. The server will respond with an SYN-ACK in response. Because of this, the attacker will not return the ACK intended. It is an interactive Mexican standoff in the digital world at its core. Until the server receives a response, it will wait. The attacker will transmit another SYN before the connection expires, and the process begins again. The attack's goal is to completely fill the connection table so that a genuine user cannot access it. When Mirai attacks its target, it uses a variety of methods. Mirai ACK is similar in fashion to Mirai SYM, which uses ACK Flooding.
- **Mirai Scan (Class_3)**: The virus and the command and control center (CnC) are the two major parts of Mirai. In addition to the malware's 10 attack vectors, Mirai also includes a scanner that actively scans out more computers to infect. Later on, the CnC plays the main initiator and can activate the single or multiple attack vectors onto the compromised devices (BOT). When the scanner is running, it randomly attempts to connect to IP addresses using the telnet protocol (on TCP port 23 or 2223). After a successful login attempt, the CnC receives the identification of the new BOT and its credentials from the CnC's database. Attackers can use the CnC's command-line interface to choose a target IP address and length of the attack. New device addresses and credentials discovered by the CnC are also used to copy over the infection code and establish further BOTs.
- **Mirai UDP (Class_5)**: The Mirai UDP attack is unique among other UDP Floods. While still a UDP Flood, the default behavior of Mirai is to randomize the source port and the destination ports. When combined with multiple source IPs (coming from multiple bots), the result is a flood of UDP traffic that can be difficult to fingerprint on an upstream router or firewall because there is no common source IP, source port, or destination port.
- **Mirai UDP plain (Class_6)**: In contrast to UDP flooding, a UDP plain attack is far more "surgical" and effective. Because of how the attacking bot "picks" ports, its effectiveness can be explained. The attacking bot will target the one that is most frequently used rather than flooding all of them. Rather than going all-in, focus the attack on a single target. This will boost the chances of success.
- **Gafgyt [Combo, Junk, Scan, TCP & UDP] (Class_7, Class_8, Class_9, Class_10 & Class_11)**: Gafgyt is an IoT botnet family that has been around for a long time with a lot of variants. A massive family with the same notoriety as Mirai has developed over time. Its variations have matured to the point that they can perform DDoS attacks, scan for vulnerabilities, execute commands, and download and execute malware in real-time. This Gafgyt service mode shows that the network of Gafgyt nodes is used for easy communication between administrators and users as well as passing command and control (C&C) instructions. Botnet administrators can use the Gafgyt network to keep track of various attack instructions supplied by users and answer inquiries and discuss ideas. Gafgyt is an Internet of Things-based botnet that uses a variety of smart routers as both bot nodes and targets. Generally, an IoT device infected with Gafgyt begins scanning the Internet for responding nodes soon after infection and then attempts penetration via weak password cracking or vulnerability exploitation, converting other PCs to bot nodes and spreading the botnet. The Gafgyt botnet prefers smart routers among IoT devices due to their vast numbers, a plethora of vulnerabilities, and weak management.

2.2. Data Pre-Processing

The traffic in the N-BaIoT dataset is unbalanced as the number of general records becomes very less with respect to attack records. Moreover, it displays the specific counts of attack for each subclass. In the N-BaIoT dataset, there are a hundred and fifteen statistically designed features retrieved from the pcap files. All these features are self-explanatory; a feature can be generalized as Header_ time-windows_ statistical variables. These features depict the following information:

- Header description
- Over five separate time-windows (100 ms; 500 ms; 1.5 s; 10 s; and 1 min)
- Seven statistical variables (mean; variance; count; magnitude; radius; covariance; and correlation coefficient) were calculated.

The decay factor value is utilized throughout the dataset and this research refers to the appropriate time window as L5, L3, L1, and so on. Using a time window, which is a preset period where information is retrieved from the traffic, makes this dataset appropriate for stateful IDS. Four information were collected from the pcap: packet count, jitter, the size of outbound packets alone, and the combined size of outbound and inbound packets. Three or more statistical measures were computed for each of these four features, resulting in a total of 23 features. In order to obtain the one hundred and fifteen characteristics in this dataset, these 23 distinct features were computed over five different time-window. The following 115 features presented in Table 1 have been used in this study. Note that some abbreviated terms are used in Table 1 which have been described in Table 2.

2.3. Feature Selection

Three variants of feature selection models are used for selecting features due to the popularity of IoT-based Intrusion Detection Systems (IDS). Those are Recursive Feature Elimination (RFE), Recursive Feature Elimination and Cross-Validation (RFECV), and Select-K-Best. In addition, RF-based feature selection has also been employed. A brief description of the algorithms is given below.

2.3.1. RF-Based Feature Selection

The relative significance of each independent variable in the RF model can also be used for feature selection. A detailed description has been discussed in Section 2.4.1.

2.3.2. Recursive Feature Elimination (RFE)

There is a lot of irrelevant features in a large dataset. The redundancy of the features affects the efficiency of the classification algorithm. As a result, the prediction may get biased. To find the effective features which are mostly responsible for expected predictions, we use RFE (Recursive Feature Elimination). This helps to reduce the dimensionality of the dataset and keep the effective features derived from the more compact subsets of features. This study employs a step-wise technique based on recursive feature elimination (RFE) to eliminate less important features. RFE is believed to be beneficial in enhancing RF models for intrusion detection [26]. It is an efficient feature selection algorithm for selecting features from a training dataset. Removing features that aren't relevant to the task can produce an input feature-set without sacrificing classification accuracy. The name "recursive" stands for the iterative approach that is deployed until a specified number of features are found. After determining the importance of each feature based on their contribution in classification using RF classifier, at first, it generates a feature ranking (high to low). Then, the least important features are removed and the model is re-trained with the revised features and the classification performance is obtained with the new feature set. This is an iterative process that continues until the feature set becomes empty.

Table 1. Name of the features used in this study.

Total Features	Feature Name
115	MI_dir_L5_weight, MI_dir_L5_mean, MI_dir_L5_variance, MI_dir_L3_weight, MI_dir_L3_mean, MI_dir_L3_variance, MI_dir_L1_weight, MI_dir_L1_mean, MI_dir_L1_variance, MI_dir_L0.1_weight, MI_dir_L0.1_mean, MI_dir_L0.1_variance, MI_dir_L0.01_weight, MI_dir_L0.01_mean, MI_dir_L0.01_variance, H_L5_weight, H_L5_mean,H_L5_variance, H_L3_weight, H_L3_mean, H_L3_variance, H_L1_weight, H_L1_mean, H_L1_variance, H_L0.1_weight, H_L0.1_mean, H_L0.1_variance, H_L0.01_weight, H_L0.01_mean, H_L0.01_variance, HH_L5_weight, HH_L5_mean, HH_L5_std, HH_L5_magnitude, HH_L5_radius, HH_L5_covariance, HH_L5_pcc, HH_L3_weight, HH_L3_mean, HH_L3_std, HH_L3_magnitude, HH_L3_radius, HH_L3_covariance, HH_L3_pcc, HH_L1_weight, HH_L1_mean, HH_L1_std, HH_L1_magnitude, HH_L1_radius, HH_L1_covariance, HH_L1_pcc, HH_L0.1_weight, HH_L0.1_mean, HH_L0.1_std, HH_L0.1_magnitude, HH_L0.1_radius, HH_L0.1_covariance, HH_L0.1_pcc, HH_L0.01_weight, HH_L0.01_mean, HH_L0.01_std, HH_L0.01_magnitude, HH_L0.01_radius, HH_L0.01_covariance, HH_L0.01_pcc, HH_jit_L5_weight, HH_jit_L5_mean, HH_jit_L5_variance, HH_jit_L3_weight, HH_jit_L3_mean, HH_jit_L3_variance, HH_jit_L1_weight, HH_jit_L1_mean, HH_jit_L1_variance, HH_jit_L0.1_weight, HH_jit_L0.1_mean, HH_jit_L0.1_variance, HH_jit_L0.01_weight, HH_jit_L0.01_mean, HH_jit_L0.01_variance, HpHp_L5_weight, HpHp_L5_mean, HpHp_L5_std, HpHp_L5_magnitude, HpHp_L5_radius, HpHp_L5_covariance, HpHp_L5_pcc, HpHp_L3_weight, HpHp_L3_mean, HpHp_L3_std, HpHp_L3_magnitude, HpHp_L3_radius, HpHp_L3_covariance, HpHp_L3_pcc, HpHp_L1_weight, HpHp_L1_mean, HpHp_L1_std, HpHp_L1_magnitude, HpHp_L1_radius, HpHp_L1_covariance, HpHp_L1_pcc, HpHp_L0.1_weight, HpHp_L0.1_mean, HpHp_L0.1_std, HpHp_L0.1_magnitude, HpHp_L0.1_radius, HpHp_L0.1_covariance, HpHp_L0.1_pcc, HpHp_L0.01_weight, HpHp_L0.01_mean, HpHp_L0.01_std, HpHp_L0.01_magnitude, HpHp_L0.01_radius, HpHp_L0.01_covariance, HpHp_L0.01_pcc

Table 2. Feature description.

Short Name	Brief Description
MI	Stats summarizing the recent traffic from this packet's Source MAC-IP
H	Stats summarizing the recent traffic from this packet's host (IP)
HH	Stats summarizing the recent traffic going from this packet's host (IP) to the packet's destination host.
HpHp	Stats summarizing the recent traffic going from this packet's host+port (IP) to the packet's destination host+port. Example: 192.168.4.2:1242 ->192.168.4.12:80
HH_jit	Stats summarizing the jitter of the traffic going from this packet's host (IP) to the packet's destination host.

Algorithm 1 depicts the steps of RFE. Figure 2 represents the flowchart. Subsets of classifier attributes are generated from all available variables and applied to the training set in an iterative manner, as mentioned above. Next, performance from the subset of classifiers is optimized; finally, a subset of classifiers with optimal results is selected.

Algorithm 1: Followed steps for Recursive Feature Elimination (RFE)

Initialization: Deploy a complete set of classifier-attributes C with the training data. Compute measurements, which are reflective of model performances. Finally, identify variable rankings.
Pre-process data
$N = len(C)$ where C = Total number of classifier-attributess
for $i = 1$ *to* N *and Subset size from classifier-attributes,* X_i **do**
    Initiate with X_i significant classifier-attributes
    Deploy X_i classifier-attributes on the training dataset
    Compute measurements, which are reflective of model performances
    Identify variable rankings
    end
Result: Classifier-attributes corresponding to optimal X_i

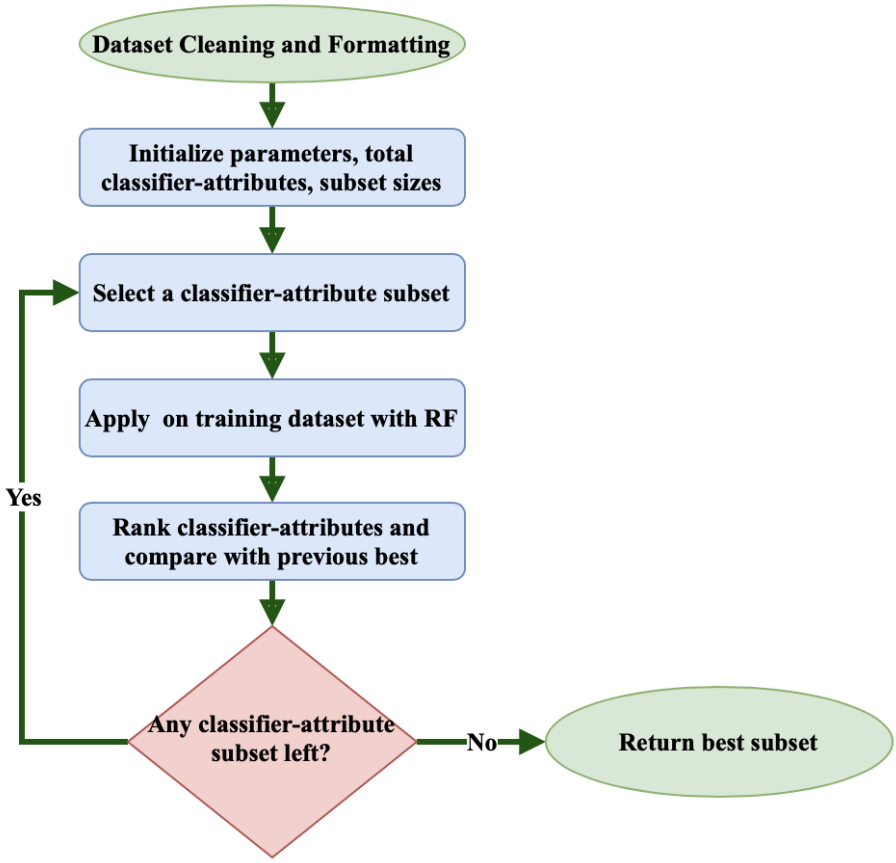


Figure 2. Workflow diagram of Recursive Feature Elimination (RFE).

2.3.3. Recursive Feature Elimination and Cross-Validation (RFECV)

If there are too many features in the raw data, applying recursive feature elimination with cross-validation (RFECV) to remove some of the unnecessary features is also very effective [29]. The best subset of features gets selected for the supplied estimator by omitting features using recursive feature elimination. The major difference between RFECV and RFE is that in RFECV, the estimator is tested in terms of generating predictions on hold-out fold data in each feature subset. As a result, the best feature subset can be identified by ranking CV scores.

2.3.4. Select-K-Best

The Select-K-Best method prioritizes top-scoring k features. The feature score is based on univariate statistical analysis, which is a one-by-one assessment of the factors. It can be applied to both classification and regression data. The score is generated for each feature. Then according to the score, the first K best scoring features with the highest scores are chosen.

2.4. Classifiers

The following section discusses the algorithms used for proposed algorithms and the efficacy of understanding the inner workings of these well-established machine learning models.

2.4.1. Random Forest (RF)

As the name suggests, the random forest (RF) algorithm is a collection of classification trees where each tree casts one vote for the most frequent class being assigned to the input data [30]. Afterward, a class with majority votes gets selected as the outcome. L. Breiman in 2001, introduced the idea on top of the Bagging (Bootstrap-Aggregating) model, which is currently known as random forest. Not only for classification and regression, but RF is also being employed for feature selection (FS) purposes. Genuer et al., in their work, contributed with the methodological model to rank variable importance through depth analysis on variable importance index [31]. Sensitivity of sample size, variable quantity, responsiveness towards different method parameters, and sensitivity towards presence of correlated variables are observed to identify RF variable importance score in their work. Being a supervised learning algorithm, random forest forms the forest, which is a part of a decision tree. It outperforms many machine learning algorithms in terms of accuracy, particularly for big datasets with numerous features. It is usually trained in the bagging method in which a randomized selection (without replacement) is performed from the training samples to construct each tree. It implies that the new trees are not reliant on older ones. Depending on the predictions of decision trees, the model aggregates its output. In general, few measurements are applied to calculate feature importance such as Gini index, mean decrease accuracy, permutation importance method (overcomes the imperfection of mean decrease impurity), etc. Due to the use of the bagging model, RF has gained significance in reducing data over-fitting phenomenon and variance among decision trees. Furthermore, handling missing data are a common treatment of RF architecture.

The RF classifier is integrated with the number of classification trees. Based on the feature importance criteria, 40 notable features are selected. The classification result is calculated using Equation (1).

$$C(t) = \max_{P} E_t \sum_{i=1}^{K} (c_i(T) = P) \tag{1}$$

where T is the training set from the original dataset S and K indicates the subsets from the T dataset. For each subset, the algorithm automatically generates K decision trees with the help of a random vector. $C(t)$ represents the classification result where $c_i(T)$ denotes the classification result of i^{th} decision tree. Here, P is the target category. However, several random forest hyper-parameters are being used to either enhance the model's prediction capability or speed up the algorithm.

RF gains higher-level performance in the case of high-dimensional data by completing an implicit FS process. Gini importance can be used within RF as measurement criteria for finding feature importance. These relevance scores help identify the decision trees significant to the classifier and can be considered as an outgrowth.

$$i(t) = 1 - f_1^2 - f_0^2 \tag{2}$$

Equation (2) is used to measure the Gini impurity. Here, each node is represented as t, which can be any node of RF decision trees. Gini impurity is used to find the optimum split, which is an estimation of measuring entropy. Furthermore, f_j in Equation (3) is the fraction of n_j samples out of the total number of samples n and j = 0,1 represents the class.

$$f_j = \frac{n_j}{n} \tag{3}$$

We can achieve decreasing δi by splitting and conveying products to two separate sub-notes (t_p & t_q) by a threshold on variable Θ. Equation (4) reflects the procedure.

$$\delta i(t) = i(t) - f_p i(t_p) - f_q i(t_q) \tag{4}$$

Next, an exhaustive search is performed with all-inclusive values of Θ, which is obtainable in the node overall thresholds. Afterward, considering all nodes t, Gini impurity values' reductions are saved for all variables separately with Equation (5). I_G reflects on the number of times feature Θ gets selected during a split and significant within the classifier during a particular problem at hand.

$$I_G(\Theta) = \sum_r \sum_q \delta i_\Theta(t, T) \tag{5}$$

2.4.2. eXtreme Gradient Boosting (XGBoost)

After selecting features using RF, an eXtreme Gradient Boosting classifier is utilized in the proposed approach to detect botnet attacks based on a few selected preconditions. XGBoost, also known as XGB, is a promising tree-based ensemble learning classifier, which is treated as the most effective implementation of gradient-boosted decision trees. Gradient boosted decision trees utilize a series of decision trees, each of which learns from the preceding tree and affects the subsequent tree; therefore, they improve the model and develop a powerful learner [32]. XGBoost combines weak classifiers to create a strong one [33]. As opposed to RF, where independent trees are generated and work individually, XGBoost incorporates feedback from previously accepted decision trees. Each iteration of gradient boosting optimizes the given loss function. The goal is to minimize the residual from the previous step where residual can be interpreted as the dissimilarity between predicted estimation and true estimation. Upon reaching a threshold point for the residual value, the final model is declared for further use. Nonetheless, if a number of decision trees fall under a threshold value before the residual can drop under the threshold, training is stopped and the final model is selected. The use of parallel execution, faster execution than gradient boosting, acceptance of regularization, etc., are a few salient features of the XGB model.

XGBoost's objective function [34] for evaluating the model's performance can be represented as Equation (6).

$$P(\theta) = t(\theta) + r(\theta) \tag{6}$$

where θ represents the parameters, r denotes the term of regularization, and t is the loss of training. Algorithm 2 depicts the XGBoost model architecture.

2.5. Model Performance Analysis

The purpose of this paper is to construct classification models using a training dataset and to assess their performance throughout the dataset. To evaluate the performance of our proposed model, several performance evaluation metrics are used. We calculate accuracy (ACC), F1 score, Kappa index, Matthew's correlation coefficient (MCC), sensitivity (SE), specificity (SP), threat score, and balanced accuracy score. Furthermore, six confusion matrices for different classifiers are also represented in Section 3.2.

Algorithm 2: Followed steps for XGBoost classifier

Input: $S \in \mathcal{R}^{n \times d}$ where data are of d dimension and n represents number of samples. Target: $F \in \mathcal{R}^{n \times 1}$

Output: $\sum_{i=1}^{C} P_i = 1, \forall i \in C = 2$ where C = the number of classes and $P \in [0, 1]$ for unrevealed test set s. P represents posterior probability

Initialization: $F_o(s) = \text{argmin}_\Omega \sum_{i=1}^{T} L(Y, \Omega)$ [33], where $L(Y, F(s))$ represents differentiable loss function and T is the total number of sampling.

for $k = 1$ *to* M $(No.\ of\ iterations = n)$ **do**

  Compute pseudo-residuals, $pr_{im} = -[\frac{\delta L(F, F(S_i))}{\delta F(S_i)}]$, where $i = 1, 2, ..., N$

  Fit a base tree, h_m deploying training set (S_i, pr_{im}) where $i = 1, 2, ..., N$

  Computing the multiplier Ω_k by $\Omega_k = \text{argmin}_\Omega \sum_{i=1}^{n} L(F_i, f_{k-1}(S_i) + \gamma h_k(S_i))$

  Upgradation of model $f_k(s) = f_{k-1}(s) + \Omega_k h_k(s)$

end

$f_k(s)$ is the target posterior probability, $P \in [0, 1]$

3. Experimental Results

The experimental results are based on the original test data. Hold out approach is used for dividing the dataset. Here, 75% of data are used for training, and the remaining 25% are used for testing purposes. The performance comparison table among classification algorithms and the confusion matrices are shown in Sections 3.1 and 3.2. In addition, evaluation on different Train-Test schemes, Receiver Operating Characteristic (ROC) Curve and comparison with other studies are demonstrated in Sections 3.3–3.5, respectively.

3.1. Performance Measures

After selecting features with RF and classifying these using XGB, some statistical measures are performed to assess the performance of the proposed method. Five other machine learning (ML) schemes have also been used: (i) RF-RF (RF classification using RF-based feature selection), (ii) RF-RFE (RF classification using RFE-based feature selection), (iii) RF-RFECV (RF classification using RFECV-based feature selection), (iv) RF-SelectK (RF classification using Select-K-best feature selection) and (v) RF-WFS (RF classification without feature selection). The classification performance is shown in Table 3.

Table 3. Performance of Classification Algorithms.

ML Schemes	ACC	F1_Score	Kappa	MCC	Sensitivity	Specificity	Threat_Score	Balanced Accuracy
RF-RF	89.6638%	86.2163%	88.5499%	89.6139%	89.6638%	98.9503%	86.3819%	94.3070%
XGB-RF	**99.9426%**	**99.9426%**	**99.9364%**	**99.9364%**	**99.9426%**	**99.9942%**	**99.8921%**	**99.9683%**
RF-RFE	89.6603%	86.2128%	88.5460%	89.6100%	89.6603%	98.9499%	86.3725%	94.3051%
RF-RFECV	89.6585%	86.2112%	88.5441%	89.6077%	89.6585%	98.9498%	86.3728%	94.3041%
RF-SelectK	89.6551%	86.2076%	88.5402%	89.6042%	89.6551%	98.9494%	86.3635%	94.3022%
RF-WFS	89.6324%	86.8542%	88.5153%	89.4106%	89.6324%	98.9472%	86.6114%	94.2898%

From Table 3, it is clear that the best accuracy (99.9426%) has been achieved from the XGB-RF model while RF-RF, RF-RFE, RF-RFECV, RF-SelectKBest, and RF-WFS model obtained nearly 90% accuracy for each. XGB-RF has also obtained the highest results for other performance metrics, including sensitivity (99.9426%), specificity (99.9942%), F1 score (99.9426%), balanced accuracy (99.9683%), etc. The proposed model has the lowest error score (0.06%). So, it can be said that the proposed model has surpassed other remaining models to a large extent in terms of performance. However, the RF-WFS approach performed worst among all of these models. In addition, we have now

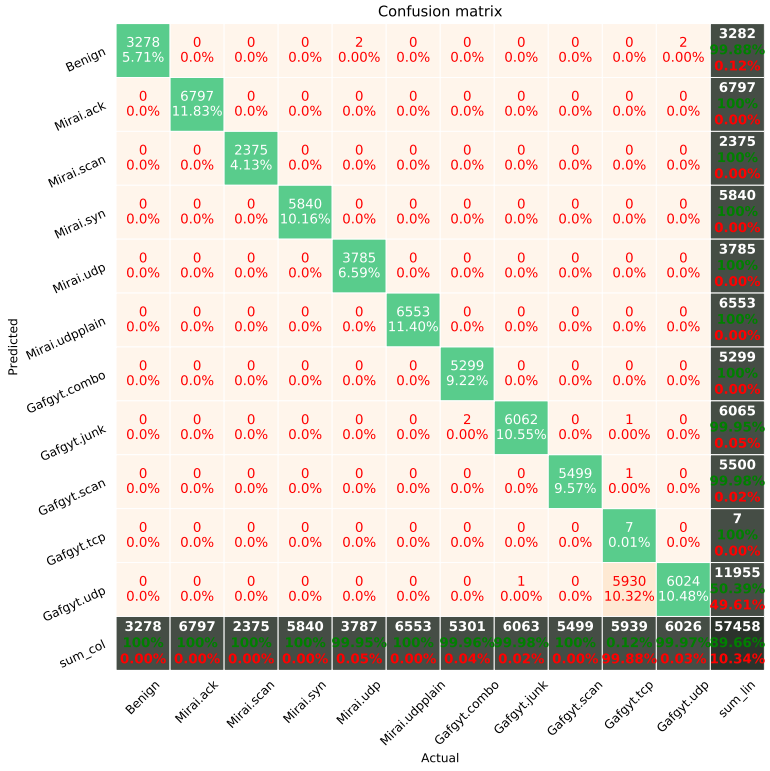
calculated the execution time of our proposed approach as well. For the whole test set (57,458 instances), it required 57.822 s in a machine having a Core i9 Processor and 64 GB of RAM. This implies that the time required for each attack detection is only 0.0010063 s using XGB-RF.

3.2. Confusion Matrix

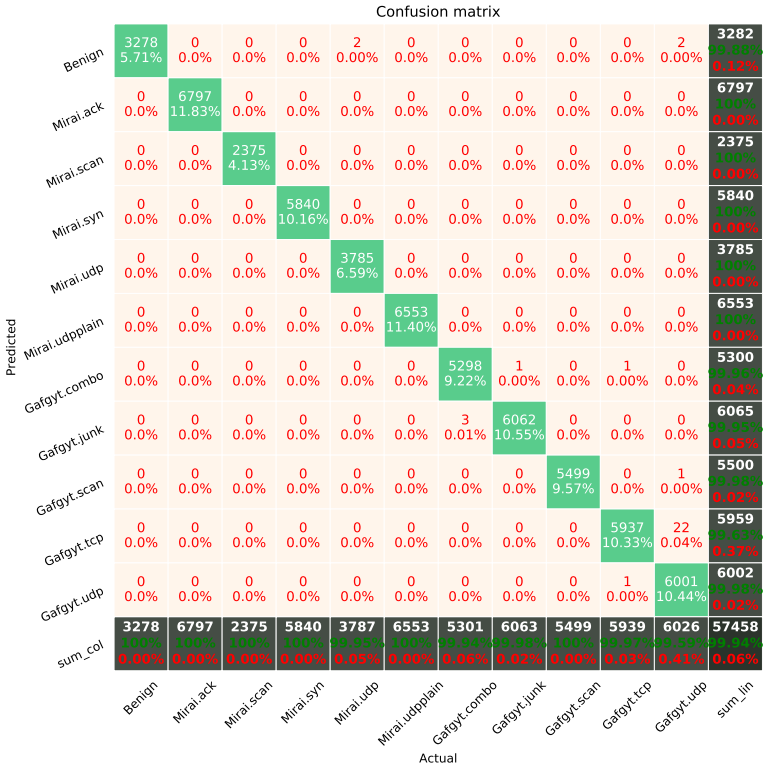
A multi-class confusion matrix is used to understand the comparativeness of different classifiers. As we have multiple classes in our N-BaIoT dataset, the multi-class confusion matrix lets us visualize the confusion faced during predicting attacks. The confusion matrix for different classifiers is shown in Figure 3.

Here, the same test set is used in all cases in Figure 3 for a fair justification. From the confusion matrix of (a) RF-RF, it is noticeable that the classifier rarely classifies Gafgyt.tcp attacks where it only classifies 7 Gafgyt.tcp attacks out of 5939. Again, in the case of Gafgyt.udp attacks, RF-RF classifies almost all the attacks. However, it misclassifies 5930 as Gafgyt.tcp which affects the accuracy score. The confusion matrix of (b) XGB-RF, which is our proposed method, overcomes the unexpected issues created by the previously discussed classifier for detecting Gafgyt.tcp and Gafgyt.udp attacks, respectively. The misclassification rate is extremely low for this XGB-RF model. The model successfully recognizes 99.9426% of attacks.

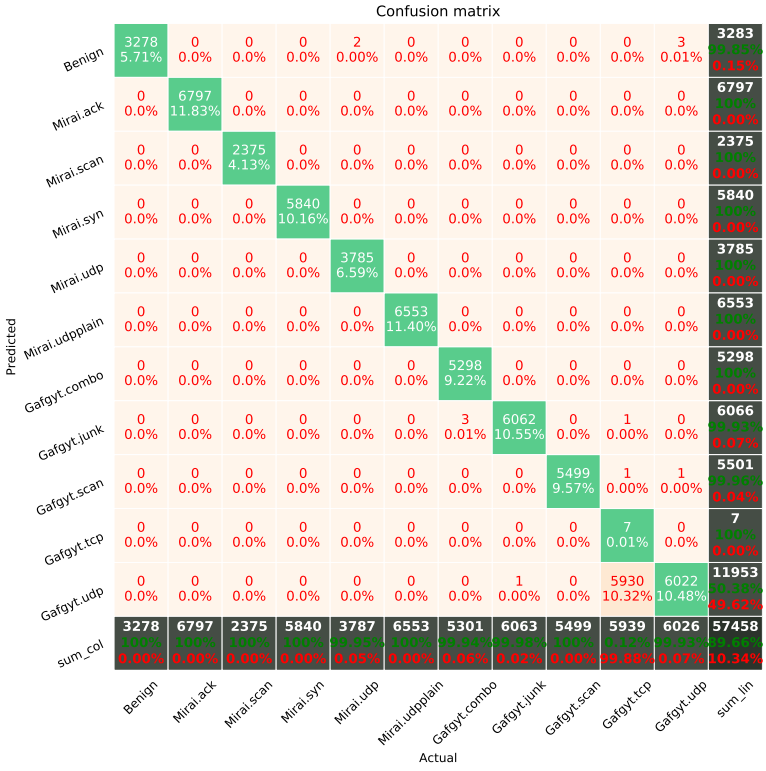
Again, from the (c) RF-RFECV confusion matrix, we see that it rarely classifies Gafgyt.tcp attacks. Here it classifies only 7 Gafgyt.tcp attacks out of 5939 while it misclassifies 5930 attacks as Gafgyt.upd. The accuracy of RF-RFECV is 89.6585% which is much lower than our proposed approach. Similarly, for (d) RF-RFE, (e) RF-SelectKBest, and (f) RF-WFS, a large amount of misclassification occurs for Gafgyt.tcp attacks as well.



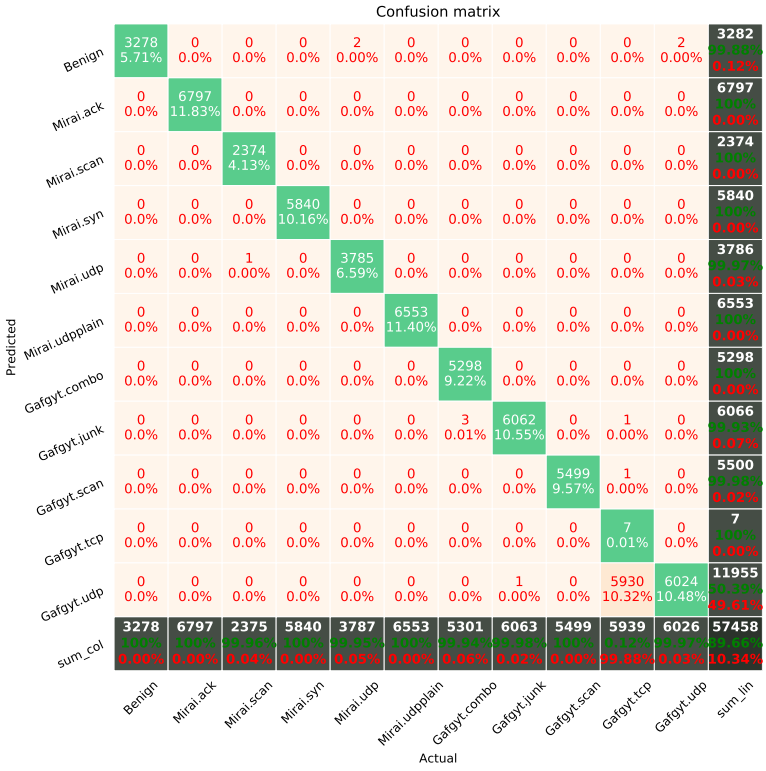
(**a**)


Confusion matrix
Predicted
Actual
Benign
Mirai.ack
Mirai.scan
Mirai.syn
Mirai.udp
Mirai.udpplain
Gafgyt.combo
Gafgyt.junk
Gafgyt.scan
Gafgyt.tcp
Gafgyt.udp
sum_col
sum_lin

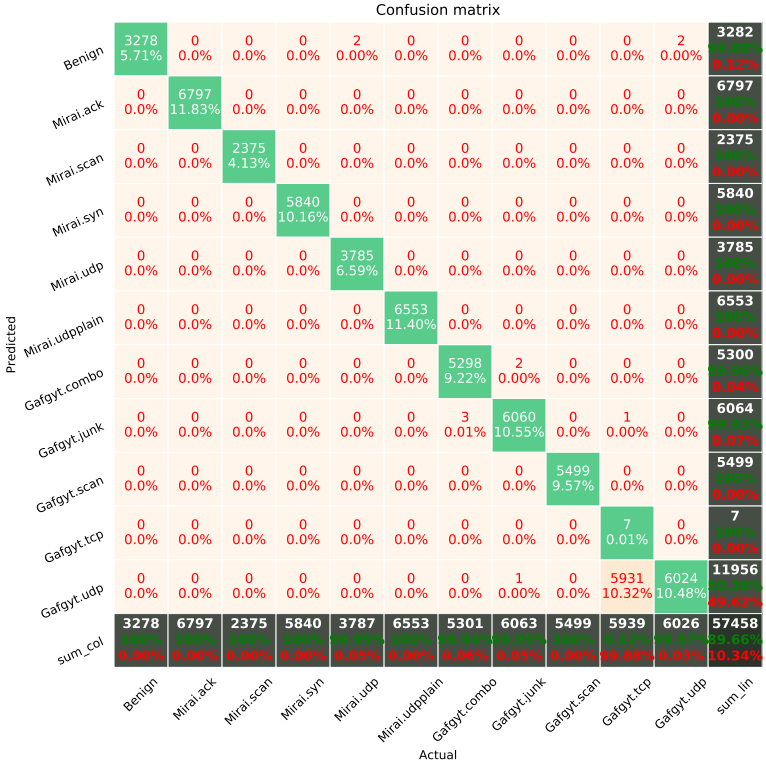
(**b**)


Confusion matrix
Predicted
Actual
Benign
Mirai.ack
Mirai.scan
Mirai.syn
Mirai.udp
Mirai.udpplain
Gafgyt.combo
Gafgyt.junk
Gafgyt.scan
Gafgyt.tcp
Gafgyt.udp
sum_col
sum_lin

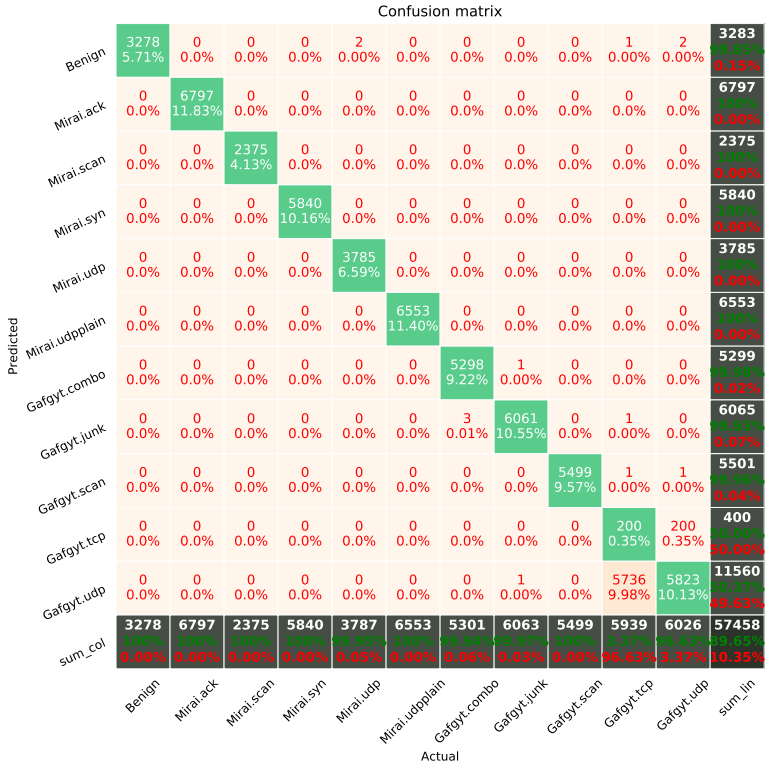
(**c**)


Confusion matrix
Predicted
Actual
Benign
Mirai.ack
Mirai.scan
Mirai.syn
Mirai.udp
Mirai.udpplain
Gafgyt.combo
Gafgyt.junk
Gafgyt.scan
Gafgyt.tcp
Gafgyt.udp
sum_col
sum_lin

(**d**)


Confusion matrix
Predicted
Actual
Benign
Mirai.ack
Mirai.scan
Mirai.syn
Mirai.udp
Mirai.udpplain
Gafgyt.combo
Gafgyt.junk
Gafgyt.scan
Gafgyt.tcp
Gafgyt.udp
sum_col
sum_lin

(**e**)



(**f**)

Figure 3. Confusion matrix for (**a**) RF-RF, (**b**) XGB-RF, (**c**) RF-RFECV, (**d**) RF-RFE, (**e**) RF-SelectKBest, (**f**) RF-WFS. Note that *X*-axis shows the actual label whereas *Y*-axis represents the predicted label by the classifier.

3.3. Evaluation on Different Train-Test Schemes

As mentioned earlier, in the proposed approach, 75% of data are used for training, and the remaining 25% are used for testing purposes. Data were primarily divided into 75–25% according to our previous experience. We also evaluate the performance using other train-test splitting schemes like 70–30% and 67–33%. These schemes are also used by other N-BaIoT studies [35,36]. The result presented in the Table 4 reveals a very minimal effect on data splitting.

Table 4. Performance Evaluation on Different Train-Test Schemes.

Performance	Train-Test (70–30%)	Train-Test (67–33%)
Accuracy	99.9564%	99.9539%
F1_Score	99.9565%	99.9539%
Kappa	99.9518%	99.9489%
MCC	99.9518%	99.9489%
Sensitivity	99.9565%	99.9539%
Specificity	99.9956%	99.9953%
Threat_Score	99.9220%	99.9160%
Balanced Accuracy	99.9760%	99.9746%

3.4. ROC Curve

It can be seen from Figure 3, that our proposed hybrid XGB-RF accurately classified all the classes, including Gafgyt.udp class. On the other hand, other methods such as RF-RF, RF-RFECV, RF-RFE, RF-SelectKBest and RF-WFS can not classify Gafgyt.udp class well. Therefore, we have generated a receiver operating characteristic curve (ROC) for Gafgyt.udp vs. all other classes in Figure 4. It can be seen that the proposed method

can recognise Gafgyt.udp than other approaches. Note that ROCs for RF-RF, RF-RFECV, RF-RFE, RF-SelectKBest and RF-WFS are overlaped together as they all exhibit the same performance. Additionally, Mason et al. [37] showed that area under ROC (AUROC) is equivalent to statistical Mann-Whitney U-statistic testing and relevant to statistical p values as we have seen in our paper that ROC of our proposed method is about 1 which is a perfect classifier and equivalently we can say $p < 0.001$. Therefore, we have not performed a statistical test or ANOVA test for statistical significance.

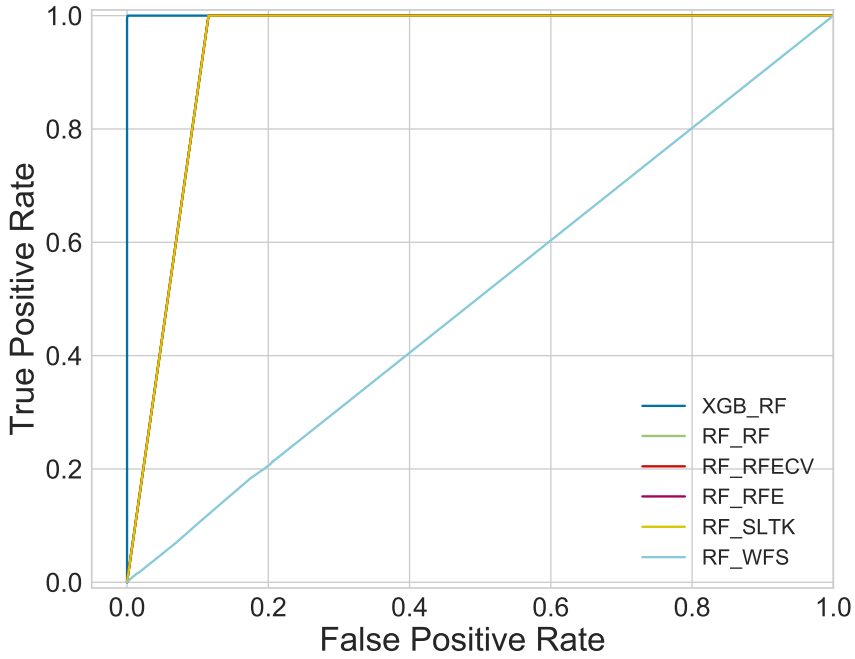


Figure 4. ROC Curve for Gafgyt.udp vs. all classes. Note that ROCs for RF-RF, RF-RFECV, RF-RFE, RF-SelectKBest, and RF-WFS are overlapped.

3.5. Performance Comparison with Other Studies

A performance comparison is performed among our proposed XGB-RF model and other contemporary studies concerning some performance indices where five of the other methods used the same N-BaIoT dataset.

Therefore, we have used those studies in the comparison. Adeel et al. In addition, Serpil et al. used RF and Deep Multilayer Perceptron (DMLP) on the same CICIDS2017 dataset and obtained an accuracy of 99.67% and 91%, respectively. Again, Kathleen et al. employed Support Vector Machine-Decision Tree-Naïve Bayes (SVM-DT-NB) classifier on the KDDCup99 dataset, which produced an accuracy of 99.62%. However, Yan et al., Chaw et al., Abdulkareem et al., Tran et al. In addition, Abdullah et al. utilized the Naïve Bayes-J48 Decision Tree-Artificial Neural Network (NB-J48-ANN)), Classification and Regression Trees (CART), Recurrent Neural Network (RNN), Collective Deep Learning and Local-Global Best Bat Algorithm for Neural Networks (LGBA-NN) approaches on the N-BaIot dataset that resulted in an accuracy of 99.10%, 99%, 89.75%, 99.84%, and 90%, respectively. Most of the above-mentioned approaches used state-of-the-art classifiers but still, their performance is less than our proposed XGB-RF approach. XGB-RF is a hybrid machine learning model that selects the prominent feature subset and improves classification accuracy. This is due to the fact that XGB-RF removes the redundant and irrelevant features. Therefore, it provides a better decision boundary that improves the classification performance and reduces the runtime. Here, from Table 5, it is clear that our proposed model outperforms all the previous approaches.

Table 5. Performance Comparison with Other Studies.

Studies	Dataset	Classifiers	Accuracy
Adeel et al. [38]	CICIDS2017	RF	99.67%
Kathleen et al. [39]	KDDCup99	SVM-DT-NB	99.62%
Yan et al. [21]	N-BaIoT	NB-J48-ANN	99.10%
Serpil et al. [26]	CICIDS2017	DMLP	91%
Chaw et al. [35]	N-BaIoT	CART	99%
Abdulkareem et al. [40]	N-BaIoT	RNN	89.75%
Tran et al. [41]	N-BaIoT	Collective Deep Learning	99.84%
Abdullah et al. [36]	N-BaIoT	LGBA-NN	90%
Proposed Method	**N-BaIoT**	**XGB-RF**	**99.94%**

4. Conclusions

The future IoT will have a profound effect on our economic and social life. Therefore, it is urgent to keep them secure. Intrusion detection systems are highly effective in identifying possible security risks and breaches. The proposed XGB-RF based hybrid machine learning scheme successfully detects different types of intrusions. Five different schemes were investigated in this proposed work. Among these, it has been shown that the accuracy, sensitivity, Kappa index of XGB-RF is almost 10% higher than other schemes. The effectiveness was tested to N-BaIoT dataset with more than 99% accuracy, which is also higher than different state-of-the-art machine learning schemes. Since the security and confidentiality of IoT devices are crucial for their success, this proposed approach can contribute a lot to enhancing the security aspects of IoT systems. However, due to the continual growth of new types of attacks, 383,379 identifying unknown attacks has proven to be a difficult task. Currently, our proposed approach takes 0.0010063 s for detecting a single attack. The near-future work could be reducing the detection time while maintaining higher accuracy so that it can be implemented in a busy IoT system. In addition, we will analyze the performance of machine learning classifiers to detect unknown attacks in IoT environments.

Author Contributions: Conceptualization, M.A.A.; methodology, J.A.F., S.T.M., J.S.T., A.S., M.M.A. In addition, M.A.A.; software, K.M.M. In addition, M.A.A.; validation, M.A.A. In addition, S.S.M.; formal analysis, K.M.M. In addition, M.A.A.; investigation, M.A.A. In addition, S.S.M.; resources, S.T.M. In addition, J.S.T.; data curation, S.T.M., J.S.T. In addition, K.M.M.; writing—original draft preparation, J.A.F., S.T.M., J.S.T., A.S. In addition, M.A.A.; writing—review and editing, M.A.A. In addition, S.S.M.; visualization, M.A.A. In addition, S.S.M.; supervision, M.A.A.; project administration, M.A.A.; funding acquisition, S.S.M. All authors have read and agreed to the published version of the manuscript.

Funding: The funding information is addressed at the Acknowledgment section.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: SSM acknowledges M1420-09-5369-FSE-000002—Post-Doctoral Fellowship, co-financed by the Madeira 14-20 Pro-gram—European Social Fund.

Conflicts of Interest: The authors declare that they have no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
IDS	Intrusion Detection System

RF	Random Forest
XGB	eXtreme Gradient Boosting
KNN	K-nearest neighbor
SVM	Support Vector Machine
RFE	Recursive Feature Elimination
RFECV	Recursive Feature Elimination and Cross-Validation
WFS	Without Feature Selection
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ACK	Acknowledge
MCC	Matthew's Correlation Coefficient
ROC	Receiver Operating Characteristic
ANN	Artificial Neural Network
NB	Naïve Bayes
DT	Decision Tree
DMLP	Deep Multilayer Perceptron
CART	Classification And Regression Trees
RNN	Recurrent Neural Network
LGBA-NN	Local-Global Best Bat Algorithm for Neural Networks

References

1. Fallahpour, A.; Wong, K.Y.; Rajoo, S.; Fathollahi-Fard, A.M.; Antucheviciene, J.; Nayeri, S. An integrated approach for a sustainable supplier selection based on Industry 4.0 concept. *Environ. Sci. Pollut. Res.* **2021**, 1–19. [CrossRef] [PubMed]
2. Attaran, M. The internet of things: Limitless opportunities for business and society. *J. Strateg. Innov. Sustain. Vol* **2017**, *12*, 11–29.
3. Symantec Internet Security Threat Report. 2019. Available online: https://docs.broadcom.com/doc/istr-24-2019-en (accessed on 30 June 2021).
4. Fruhlinger, J. Top Cybersecurity Facts, Figures and Statistics. 2020. Available online: https://www.csoonline.com/article/3153707/top-cybersecurity-facts-figures-and-statistics.html (accessed on 30 June 2021).
5. A Perfect Storm: The Security Challenges of Coronavirus Threats and Mass Remote Working. 2020. Available online: https://blog.checkpoint.com/2020/04/07/a-perfect-storm-the-security-challenges-of-coronavirus-threats-and-mass-remote-working/ (accessed on 30 June 2021).
6. Manimurugan, S.; Al-Mutairi, S.; Aborokbah, M.M.; Chilamkurti, N.; Ganesan, S.; Patan, R. Effective attack detection in internet of medical things smart environment using a deep belief neural network. *IEEE Access* **2020**, *8*, 77396–77404. [CrossRef]
7. Fathollahi-Fard, A.M.; Ahmadi, A.; Karimi, B. Multi-Objective Optimization of Home Healthcare with Working-Time Balancing and Care Continuity. *Sustainability* **2021**, *13*, 12431. [CrossRef]
8. Muthanna, M.S.A.; Muthanna, A.; Rafiq, A.; Hammoudeh, M.; Alkanhel, R.; Lynch, S.; Abd El-Latif, A.A. Deep reinforcement learning based transmission policy enforcement and multi-hop routing in QoS aware LoRa IoT networks. *Comput. Commun.* **2021**, *183*, 33–50. [CrossRef]
9. Fathollahi-Fard, A.M.; Dulebenets, M.A.; Hajiaghaei-Keshteli, M.; Tavakkoli-Moghaddam, R.; Safaeian, M.; Mirzahosseinian, H. Two hybrid meta-heuristic algorithms for a dual-channel closed-loop supply chain network design problem in the tire industry under uncertainty. *Adv. Eng. Inform.* **2021**, *50*, 101418. [CrossRef]
10. Moosavi, J.; Naeni, L.M.; Fathollahi-Fard, A.M.; Fiore, U. Blockchain in supply chain management: A review, bibliometric, and network analysis. *Environ. Sci. Pollut. Res.* **2021**, *5*, 1–15. [CrossRef]
11. Rafiq, A.; Ping, W.; Min, W.; Muthanna, M.S.A. Fog Assisted 6TiSCH Tri-Layer Network Architecture for Adaptive Scheduling and Energy-Efficient Offloading Using Rank-Based Q-Learning in Smart Industries. *IEEE Sens. J.* **2021**, *21*, 25489–25507. [CrossRef]
12. Marzano, A.; Alexander, D.; Fonseca, O.; Fazzion, E.; Hoepers, C.; Steding-Jessen, K.; Chaves, M.H.; Cunha, Í.; Guedes, D.; Meira, W. The evolution of bashlite and mirai iot botnets. In Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 25–28 June 2018; pp. 00813–00818.
13. Cisco Annual Internet Report (2018–2023) White Paper. 2020. Available online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed on 30 June 2021).
14. Vasilomanolakis, E.; Karuppayah, S.; Mühlhäuser, M.; Fischer, M. Taxonomy and survey of collaborative intrusion detection. *Acm Comput. Surv.* **2015**, *47*, 1–33. [CrossRef]
15. Summerville, D.H.; Zach, K.M.; Chen, Y. Ultra-lightweight deep packet anomaly detection for Internet of Things devices. In Proceedings of the 2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC), Nanjing, China, 14–16 December 2015; pp. 1–8.
16. Midi, D.; Rullo, A.; Mudgerikar, A.; Bertino, E. Kalis—A system for knowledge-driven adaptable intrusion detection for the Internet of Things. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 656–666.

17. Alothman, Z.; Alkasassbeh, M.; Al-Haj Baddar, S. An efficient approach to detect IoT botnet attacks using machine learning. *J. High Speed Netw.* **2020**, *26*, 241–254. [CrossRef]
18. Aburomman, A.A.; Reaz, M.B.I. Review of IDS development methods in machine learning. *Int. J. Electr. Comput. Eng.* **2016**, *6*, 2432–2436.
19. Bijalwan, A. Botnet forensic analysis using machine learning. *Secur. Commun. Netw.* **2020**, *2020*, 9302318. [CrossRef]
20. Shafiq, M.; Tian, Z.; Sun, Y.; Du, X.; Guizani, M. Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city. *Future Gener. Comput. Syst.* **2020**, *107*, 433–442. [CrossRef]
21. Soe, Y.N.; Feng, Y.; Santosa, P.I.; Hartanto, R.; Sakurai, K. Machine learning-based IoT-botnet attack detection with sequential architecture. *Sensors* **2020**, *20*, 4372. [CrossRef]
22. Diro, A.A.; Chilamkurti, N. Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Gener. Comput. Syst.* **2018**, *82*, 761–768. [CrossRef]
23. Ahmad, I.; Basheri, M.; Iqbal, M.J.; Rahim, A. Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. *IEEE Access* **2018**, *6*, 33789–33795. [CrossRef]
24. Deng, L.; Li, D.; Yao, X.; Cox, D.; Wang, H. Mobile network intrusion detection for IoT system based on transfer learning algorithm. *Clust. Comput.* **2019**, *22*, 9889–9904. [CrossRef]
25. Mirsky, Y.; Doitshman, T.; Elovici, Y.; Shabtai, A. Kitsune: An ensemble of autoencoders for online network intrusion detection. *arXiv* **2018**, arXiv:1802.09089.
26. Ustebay, S.; Turgut, Z.; Aydin, M.A. Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier. In Proceedings of the 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), Ankara, Turkey, 3–4 December 2018; pp. 71–76.
27. da Costa, K.A.; Papa, J.P.; Lisboa, C.O.; Munoz, R.; de Albuquerque, V.H.C. Internet of Things: A survey on machine learning-based intrusion detection approaches. *Comput. Netw.* **2019**, *151*, 147–157. [CrossRef]
28. Meidan, Y.; Bohadana, M.; Mathov, Y.; Mirsky, Y.; Shabtai, A.; Breitenbacher, D.; Elovici, Y. N-baiot—Network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Comput.* **2018**, *17*, 12–22. [CrossRef]
29. Xie, H.; Wei, S.; Zhang, L.; Ng, B.; Pan, S. Using feature selection techniques to determine best feature subset in prediction of window behaviour. In Proceedings of 10th Windsor Conference: Rethinking Comfort. NCEUB, Windsor, UK, 12–13 April 2018; pp. 315–328.
30. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
31. Genuer, R.; Poggi, J.M.; Tuleau-Malot, C. Variable selection using random forests. *Pattern Recognit. Lett.* **2010**, *31*, 2225–2236. [CrossRef]
32. Parsa, A.B.; Movahedi, A.; Taghipour, H.; Derrible, S.; Mohammadian, A.K. Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis. *Accid. Anal. Prev.* **2020**, *136*, 105405. [CrossRef] [PubMed]
33. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [CrossRef]
34. Awal, M.A.; Masud, M.; Hossain, M.S.; Bulbul, A.A.M.; Mahmud, S.H.; Bairagi, A.K. A novel bayesian optimization-based machine learning framework for COVID-19 detection from inpatient facility data. *IEEE Access* **2021**, *9*, 10263–10281. [CrossRef]
35. Htwe, C.S.; Thant, Y.M.; Thwin, M.M.S. Botnets Attack Detection Using Machine Learning Approach for IoT Environment. *J. Phys. Conf. Ser.* **2020**, *1646*, 012101. [CrossRef]
36. Alharbi, A.; Alosaimi, W.; Alyami, H.; Rauf, H.T.; Damaševičius, R. Botnet Attack Detection Using Local Global Best Bat Algorithm for Industrial Internet of Things. *Electronics* **2021**, *10*, 1341. [CrossRef]
37. Mason, S.J.; Graham, N.E. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation. *Q. J. R. Meteorol. Soc. A J. Atmos. Sci. Appl. Meteorol. Phys. Oceanogr.* **2002**, *128*, 2145–2166. [CrossRef]
38. Abbas, A.; Khan, M.A.; Latif, S.; Ajaz, M.; Shah, A.A.; Ahmad, J. A New Ensemble-Based Intrusion Detection System for Internet of Things. *Arab. J. Sci. Eng.* **2021**, 1–15. [CrossRef]
39. Goeschel, K. Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive Bayes for off-line analysis. In Proceedings of the SoutheastCon 2016, Norfolk, VA, USA, 30 March–3 April 2016; pp. 1–6. [CrossRef]
40. Hezam, A.A.; Mostafa, S.A.; Ramli, A.A.; Mahdin, H.; Khalaf, B.A. Deep Learning Approach for Detecting Botnet Attacks in IoT Environment of Multiple and Heterogeneous Sensors. In Proceedings of the International Conference on Advances in Cyber Security, Penang, Malaysia, 24–25 August 2021; Springer: Singapore, 2021; pp. 317–328.
41. Khoa, T.V.; Saputra, Y.M.; Hoang, D.T.; Trung, N.L.; Nguyen, D.; Ha, N.V.; Dutkiewicz, E. Collaborative Learning Model for Cyberattack Detection Systems in IoT Industry 4.0. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC), Seoul, Korea, 25–28 May 2020; pp. 1–6. [CrossRef]