Journal of
Sensor and
Actuator Networks

MDPI

Article

ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks

Qasem Abu Al-Haija * and Mu'awya Al-Dala'ien

Department of Computer Science/Cybersecurity, Princess Sumaya University for Technology (PSUT), Amman 11941, Jordan; m.aldalaien@psut.edu.jo
* Correspondence: q.abualhaija@psut.edu.jo

Abstract: Due to the prompt expansion and development of intelligent systems and autonomous, energy-aware sensing devices, the Internet of Things (IoT) has remarkably grown and obstructed nearly all applications in our daily life. However, constraints in computation, storage, and communication capabilities of IoT devices has led to an increase in IoT-based botnet attacks. To mitigate this threat, there is a need for a lightweight and anomaly-based detection system that can build profiles for normal and malicious activities over IoT networks. In this paper, we propose an ensemble learning model for botnet attack detection in IoT networks called ELBA-IoT that profiles behavior features of IoT networks and uses ensemble learning to identify anomalous network traffic from compromised IoT devices. In addition, our IoT-based botnet detection approach characterizes the evaluation of three different machine learning techniques that belong to decision tree techniques (AdaBoosted, RUSBoosted, and bagged). To evaluate ELBA-IoT, we used the N-BaIoT-2021 dataset, which comprises records of both normal IoT network traffic and botnet attack traffic of infected IoT devices. The experimental results demonstrate that our proposed ELBA-IoT can detect the botnet attacks launched from the compromised IoT devices with high detection accuracy (99.6%) and low inference overhead (40 μ-seconds). We also contrast ELBA-IoT results with other state-of-the-art results and demonstrate that ELBA-IoT is superior.

Keywords: Internet of Things (IoT); intrusion detection system (IDS); machine learning; ensemble learning; botnet attacks; anomaly detection

check for updates

Citation: Abu Al-Haija, Q.; Al-Dala'ien, M. ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks. *J. Sens. Actuator Netw.* **2022**, *11*, 18. https://doi.org/10.3390/jsan11010018

Academic Editor: Lei Shu

Received: 11 February 2022
Accepted: 8 March 2022
Published: 9 March 2022

1. Introduction

Internet of Things (IoT) is one of the most emerging paradigms in the networking realm. It can be defined as the "interconnection of things" having constrained computational power and capabilities. It can be used to send and receive data over the internet without requiring human-to-computer or human-to-human interaction [1]. The word "things" refers to the IP-enabled, networked devices (both physical and virtual). Things may include telematics boxes, self-driving cars, printers, surveillance cameras, tablets, smartphones, ultra-wideband (UWB), infrared data association (IrDA), ZigBee, NFC data centers, and cellular and Wi-Fi networks. The IoT with all its subtechnologies is considered a network of numerous physical objects (24.15 billion devices in 2019 jumping to 76.45 billion computing devices in 2026) [2]. The financial impact of the IoT may be from $3.9 to $11.1 trillion on the global economy by 2025 [3].

In addition to Internet Protocol (IP), these devices are enabled with several other important technologies, including radio-frequency identification (RFID) technology, sensors, actuators, GPS services, nanotechnologies, near field communication (NFC), and cloud computing.

Apart from the traditional IoT, resource-constrained IoT devices are also worth mentioning. Although these devices involve IoT applications, they are small, low-power, battery-operated devices with disparate design trade-offs. In addition, they also have limited storage and computational capabilities. In a nutshell, resource-constrained devices are endnotes with sensors used for handling particular application purposes. These devices are resource-limited in various aspects, such as limited storage and processing abilities and limited energy due to energy-prone batteries. In addition, they are connected via low-power lossy links, via vulnerable radio conditions, and without human interaction [4].

Despite all the aforementioned facts about the IoT, cybersecurity vulnerabilities are common in the IoT. Exploiting these security vulnerabilities, hackers can establish a botnet and execute commands either remotely or locally. They can also gain unauthorized access and modify sensitive data, disrupt normal operations of the IoT, or damage the IoT altogether. The vulnerability can exist in both hardware and software components of the IoT [5]. Hardware vulnerabilities are difficult to detect and much harder to fix due to various embedded microprograms in them. The reasons why hardware vulnerabilities cannot easily be fixed include lack of expertise, cost, interoperability, and incompatibility. Contrarily, software vulnerabilities exist in the software components of the IoT, such as OSec, communication protocols, and other applications. According to TechRadar, an IT security firm, the number of threats against Internet of Things (IoT) gadgets and smartphones increased quickly in 2018 [6]. McAfee also believes that malware attacks on IoT gadgets will continue to occur, as more than 25 million smart speakers or voice assistants are already in use.

In this work, we specifically pay heed to understanding botnet attack detection in IoT networks. Before that, we comprehend the concept of the botnet. A botnet is a collection of thousands or even millions of infected computers, each of them called a bot or zombie. In simple words, millions of bots collectively form a botnet, which is remotely controlled by botmasters using a command and control (C&C) server. As per AVG (Anti-Virus-Guard) Technologies, "at their most basic, botnets are made up of large networks of "zombie" computers all obeying one master computer" [7]. Bot detection and response have become a Gordian knot for today's information security defense systems. Bot herders are significantly evolving their botnet propagation and C&C technologies to evade the latest botnet detection techniques from IT security folks. The following figure, Figure 1, shows how a botmaster (a type of hacker) establishes a botnet network using a C&C server. Several bots which can be seen in the figure can provide sensitive information to the botmaster. Each bot can be assumed a separate PC in the organization that includes private information.

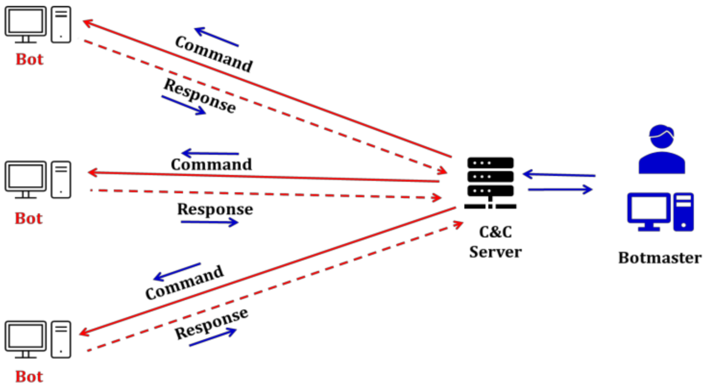


Figure 1. Botnet operation via C&C server.

In this paper, we make use of an ensemble learning model to detect botnet attacks in IoT networks. Ensemble learning involves multiple algorithms to obtain better predictive performance, rather than using a single algorithm of its constituent algorithms [8]. Moreover, the ensemble learning model can deal with data heterogeneity or class imbalance problems that are often faced during anomaly detection [9]. This paper is based on the N-BaIoT2021 dataset [10], which is a network-based anomaly detection approach in ensemble learning. N-BaloT2021 identifies abnormal network behavior to detect anomalous network traffic from the compromised IoT [10]. The N-BaIoT2021 dataset includes different botnet attacks belonging to the two most common IoT botnet vectors, Bashlite and Mirai. The dataset was prepared for comparative analysis of anomaly detection using five IoT devices: a doorbell, thermostat, baby monitor, security camera, and webcam. In addition to the 611,359 samples of normal traffic, the N-BaIoT2021 dataset includes 7737 samples of botnet traffic (malicious) distributed as follows: Bashlite (4737 samples) and Mirai (3000 samples) against the five mentioned IoT devices. Specifically, our contributions in this paper can be stated as follows:

- We present a comprehensive efficient detection/classification model/architecture with detailed preprocessing operations that can classify the IoT traffic records of the N-BaIoT2021 dataset into binary classifier (2-class), ternary classifier (3-class), and multiclassifier (10-class).
- We characterize the performance of four machine-learning-based decision tree models: AdaBoosted decision tree, RUSBoosted decision tree, bagged decision tree, and their ensemble learning model.
- We provide an inclusive experimental evaluation to gain more insight into the system efficiency and solution approaches, such as the confusion matrix, model precision, model sensitivity, and others.
- We contrast our best performance results with state-of-the-art works employing several supervised learning algorithms in the same area of study.

The remaining part of this paper is structured as follows: Section 2 provides a literature review with a summary of the most recent surveyed papers. Section 3 presents the system model and architecture with a detailed explanation for each underlying subsystem. Section 4 reports extensive experimental outcomes and performance trajectory as well as a comparison with existing models. Finally, Section 5 concludes the presented work.

2. Related Work

A significant number of studies have been conducted to understand the nature of IoT cyber-attacks. IoT security professionals have used various detection methods and techniques to detect attacks on IoT networks. In the meantime, attackers have always been sophisticated and fast in their operations. These attackers also developed the latest attacking techniques to counter safeguarding mechanisms. Much research has been conducted to accurately detect anomalies in the IoT. Most IoT-based studies reveal botnet attacks that are based on command and control (C&C) servers. In this scenario, zombies or bots, infected computer s, act as clients to communicate with a C&C server that passes instructions to each zombie computer. If the control of the central server is gained, the entire botnet can be broken down. This is also known as client–server model. Another approach used by hackers against the IoT is the peer-to-peer model. Instead of connecting with a central server, zombie computers in this model are connected via the internet. This model comes into place to fix weaknesses of the client–server model, as removing one or more bots will not ease the problem due to the huge botnet of independent bots. Several IDSes were proposed and developed during the last couple of years; examples can be found in [11–15].

Previous studies also shed light on the types of botnet attacks that hackers used in the face of the IoT. For example, dial-up bots compromise dial-up modems to force them to dial phone numbers for malicious purposes. A "click fraud" is an illegal act of clicking on pay per click (PPC) advertisements or banner ads to increase the number of clicks for advertisers or bot herders. They design automated online bots to create click frauds. Spyware can be used to secretly gather sensitive data, such as credit card numbers or login details, in the victim's IoT device. They can be beneficial for botmasters because they can sell such data on the black market. In addition, a DoS attack is used to compromise a target, such as a server, using multiple compromised computers (botnet) to cause a denial of service for the victim. Botnets are widely used to launch massive DoS and DDoS attacks. The Mirai botnet was utilized in 2016 to attack the domain name service provider Dyn, based in Manchester, and attack volumes were gauged at over 600 Gbps. Lastly, a spambot is a malicious machine that is used to distribute automated spam emails.

Botnet detection techniques for the IoT are either network-based or host-based [16–20]. However, the host-based approach is less realistic. For instance, in [16,17], the authors developed a comprehensive architecture for IoT instruction detection and classification at the network layer of the IoT paradigm. Six different supervised ML methods were employed to develop the IoT-IDS: three ensemble learning methods, two neural network methods, and one kernel method. They evaluated their models using two well-known IoT attack datasets, distilled-Kitsune-2018 and NSL-KDD. Their best results were better than any prior art by 1~20%. In [18], the authors proposed port scanning attack detection for IoT networks by characterizing the performance of several machine learning techniques. However, their best empirical results were recorded for the best logistic regression model, which scored 99.4% and 99.7%, registered for accuracy and F-score with low detection overhead. In [19–22], the authors' studies barely used the latest techniques, such as autoencoders, which are a part of our proposed technique in ensemble learning—known as N-BaIoT. This means that IoT attacks often remained undetected and unnoticed due to a lack of modern security defense mechanisms. However, now, autoencoders are being used in this paper to provide more accurate and useful results. Autoencoders are mostly used for anomaly detection in wireless sensor networks (WSNs) that are embedded in IoT devices. The algorithm used for detection purposes involves two components—one is located in the IoT cloud, while the other is placed within sensors. During the evaluation, it was seen that the autoencoders' unsupervised learning features enabled adaptation to unexpected changes in IoT network environments. Attempts have also been made to compare many deep learning models for network intrusion detection (NID), including the recurrent neural network (RNN), self-taught learning (STL), and vanilla deep neural net (DNN). Machine learning (ML)-based autoencoders provided efficient outcomes, as they met even the unique constraints of the IoT environments. These constraints included small storage capacity and shortage of computing power due to the IoT's tiny size. Nevertheless, autoencoders provided useful results in detecting botnet attacks. More importantly, some previous studies also relied on statistical approaches that only worked on some predefined values. Unfortunately, cybersecurity threats and attacks on the IoT are accelerating by leaps and bounds, and bot herders are developing the latest attacking techniques. Therefore, statistical approaches could not work against the latest techniques. This is the reason the researcher preferred behavioral-based N-BaIoT, which is dynamic and efficient and provided more accurate results. The proposed method has several advantages. For example, it uses online processing, so it is deployable even in low-memory IoT. This ensemble learning approach is scalable and lightweight across various IoT devices. Furthermore, detection accuracy is very high, and computational load on devices is very low.

The number of false positives is also negligible. Moreover, the unsupervised learning feature facilitates adaptability to dynamic environments.

L. Yang et al. [23] proposed an adaptive IoT streaming data analytics framework—namely, "LightGBM and Drift Adaptation"—to detect anomalies in IoT data streams without the involvement of human beings. R. Qaddoura et al. [24] recommended an approach that was based on three stages: the reduction stage, the support vector machine and synthetic minority oversampling technique (SVM-SMOTE), and the single hidden layer feed-forward neural network (SLFN) stage. The outcomes of this research demonstrated that the SLFN technique and SVM-SMOTE with a ratio of 0.9 and the k value of 3 for the k-means++ clustering method gave better results than other classification techniques and other values. Another study conducted by Wan-Chen Shi et al. [25] showed that the behaviors of network traffic from network packets were inspected through DeepBot, a deep learning method, to detect botnets and classify them into disparate categories. Huy-Trung Nguyen et al. [26] also conducted research for Linux IoT botnet detection. It was based on PSI graph and DGCNN classifiers. In this experiment, the researchers used

10033 EFL files that further included 6031 benign files and 4002 IoT botnet samples. The results showed that PSI graph and DGCNN classifiers achieved an F-measure of 94% and an accuracy of 92%. In addition to the aforementioned studies, the researcher added another work in this paper. Christopher D. McDermott et al. [27] used the BLSTM-RNN detection model to detect botnets within consumer-based IoT devices and networks. Researchers used the word embedding technique to recognize text and convert attack packets into a tokenized integer format. They detected four attack vectors used by Mirai botnet malware and evaluated them for loss and accuracy. According to researchers of this study, the bidirectional technique added overhead to each epoch and increased processing time. However, it proved to be a better progressive model over time. Finally, authors in [28] proposed a detection system for a Ping flood attack in an Internet of Things (IoT) network using a k-means algorithm. Their proposed system exhibits high accuracy results, scoring a 99.94%, with very low false negatives (0.00%) and false positives (1.38%). To sum up, Table 1 summarizes some facts and figures about some recent research in this regard below.

Table 1. Summary of reviewed related work research from the last five years.

Paper	Attack(s)	Detection Approach	Deployment Level	Assumed Environment	Data for Evaluation
[11]	DoS attacks, probe attacks, root-to-local (r2l) attacks, user-to-root (u2r) attacks	CNN	Layered approach	Platform independent	NSL-KDD dataset
[12]	OS Scan attack, Fuzzing attack, video injection, ARP attack, active wiretap attack, SSDP flood attack, SYN DoS attack, SSL renegotiation attack, and Mirai attack	Ensemble-learning-based Boosted Trees	Network	-	Distilled-Kitsune-2018 dataset
[13]	Port scan attack	Logistic regression	Network	Windows environment	Port scanning dataset 2017
[14]	-	DeL-IoT	Network	-	-
[15]	DoS attack, spam, and data theft	Ensemble learning	Network	-	Network traffic
[16]	Mirai-infected IoT devices scan for further devices	Dynamic updating of flow rules	"Thin fog"	Critical infrastructures	Emulated IoT nodes, simulated data
[17]	Worm propagation, code injection, tunneling attack	Deep learning	Host	-	Two real devices
[18]	-	LightGBM model and drift adaptation	-	-	IoT data streams
[19]	-	SLFN, SVM-SMOTE	Network	-	IoT network data
[20]	-	DeepBot-deep learning model	Network	-	Network packets
[21]	Botnet IoT malware scan	PSI graph and CNN classifier	Host	Linux environment	10033 ELF (4002 IoT botnet samples and 6031 benign files)
[22]	IoT-based DDoS	BLSTM-RNN detection model	Host/ Network	-	-
[23]	Ping flood attacks	K-Means	Network	-	IoT network data

3. ELBA-IoT Model Development

ELBA-IoT is an anomaly-based intrusion detection system developed to autonomously build profiles for normal and malicious (botnets') behaviors and then provide detection/classification for the traffic communicated through the IoT network. In this research, the proposed system is decomposed into three distinct modules, where each module is implemented via a number of operations. These modules are responsible for processing the input (N-BaIoT 2021 dataset) through a series of consecutive operations to provide the final output (anomaly-based detection). Specifically, the proposed ELBA-IoT system is composed of a data preparation (DP) module, learning process (LP) module, and evaluation process (EP) module, as illustrated in Figure 2.

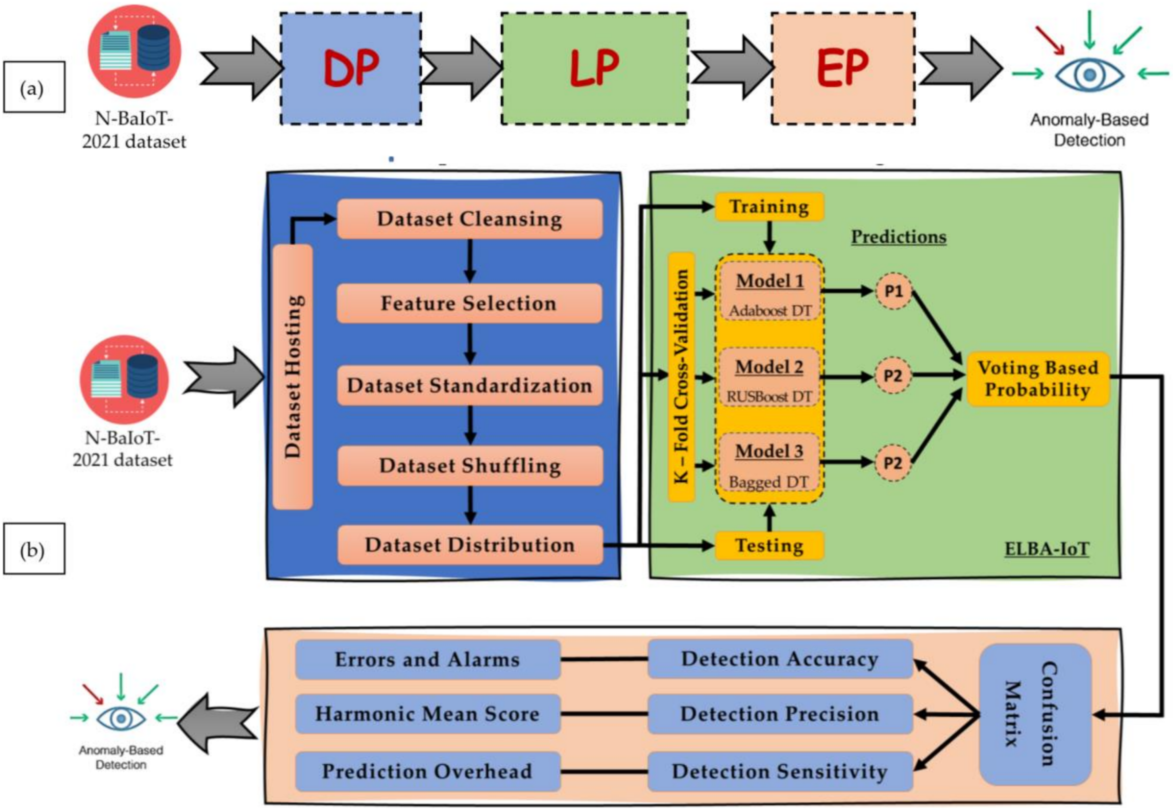


Figure 2. The three main modules of the proposed ELBA-IoT system: (**a**) top view diagram, (**b**) detailed architecture. DP: data preparation, LP: learning process, EP: evaluation process.

3.1. Implementation of the Data Preparation (DP) Module

The data preparation (DP) module concerns the preprocessing operations for the raw IoT traffic data of the N-BaIoT 2021 dataset and the transformation of preprocessed data into a table of labeled features that can be fed into and trained by the machine learning part of the LP module. The implementation phases of this module include the following consecutive operations:

3.1.1. Data Hosting Process (DHP)

The data hosting process (DHP) is the process of keeping the data on a stable and accessible platform that remains persistent and highly reliable. In this research, we use the MATLAB computing platform as a system to host, train, and evaluate the data and the model. Therefore, this stage is responsible for accepting the collected data records in its CSV format (comma-separated values) and importing the data using MATLAB tables that can be used for further preprocessing operations. By this hosting, every IoT traffic record is formatted as raw in the table, while the data features are represented as columns.

3.1.2. Data Cleansing Process (DCP)

The data cleansing process (DHP) is the process of exploring the dataset to obtain a deeper understanding of the underlying dataset and provide correction for the misinterpreted data. DHP deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data [29]. In this research, we conducted several DHP processes over the imported data, including missing value checks (searching for null-value cells and replacing with zero numerical), corrupted value checks (searching for misinterpreted data and removing them), fixing the attribute names for the main features (the imported data from CSV usually have no names for their attributes), maintaining an atomic representation of the data (make sure all attributes are simple and filled with a single value for each cell), duplicate data checks (make sure all data samples are respected once with no redundancy of specific data records), and label encoding (the target feature is indeed encoded using integer encoding). The output classes are given the values 0 and 1 for the binary classifier and values from 0 to 9 for the multiclassifier. Table 2 below summarizes the encoding process for the target labels and fixes all typos and incorrect data records.

Table 2. Label encoding for the target classes (output labels).

Classifier	Normal	Botnet(s)
Binary Classifier	0 (normal)	1 (anomaly)
Ternary Classifier	0 (normal)	1. Mirai botnet 2. Bashlite botnet (Gafgyt)
Multiclassifier	0 (normal)	1. MIRAI_DANMINI_DOORBELL 2. MIRAI_ECOBEE_THERMOSTAT 3. MIRAI_PHILIPS_BABY_MONITOR 4. MIRAI_PROVISION_SECURITY_CAMERA 5. MIRAI _SAMSUNG_WEBCAM 6. GAFGYT_DANMINI_DOORBELL 7. GAFGYT_ECOBEE_THERMOSTAT 8. GAFGYT_PHILIPS_BABY_MONITOR 9. GAFGYT_PROVISION_SECURITY_CAMERA

3.1.3. Feature Selection Process (FSP)

In feature selection, we aim to select the features which are highly dependent on the response. The feature selection process (FSP) is the process of selecting all features that positively contribute to the performance of the machine learning model and avoid other features that may negatively impact the model performance. In this study, we employed the correlation coefficient score approach (CCS) to define the most prominent features to be used for ELBA-IoT. The CCS method chooses all the possible feature combinations and then calculates the linear relationship between features and the target. The logic behind using correlation for feature selection is that the good features are highly correlated with the target. Accordingly, the CCS method finds the best features that can be used later in developing the ML model. Consequently, the computationally intensive problem can be overcome by running the algorithm with fewer features for detecting botnet attacks. The Pearson correlation coefficient score [30] used in this research is given by the following formula:

$$\rho_{X,\,Y} = \frac{cov\ (X,\ Y)}{\sigma_X\sigma_Y}\ where:\ cov\ (X,\ Y) = \frac{1}{N}\sum_{i=1}^{N}(X - mean\ (x))(Y - mean(Y))$$

where Pearson's correlation coefficient ($\rho_{X,\,Y}$) is the covariance of two variables divided $cov\ (X,\ Y)$ by the product of their standard deviations ($\sigma_X\sigma_Y$). It is valued between -1 and 1, negative values meaning inverse relation and positive meaning the reverse case.

3.1.4. Data Standardization Process (DSP)

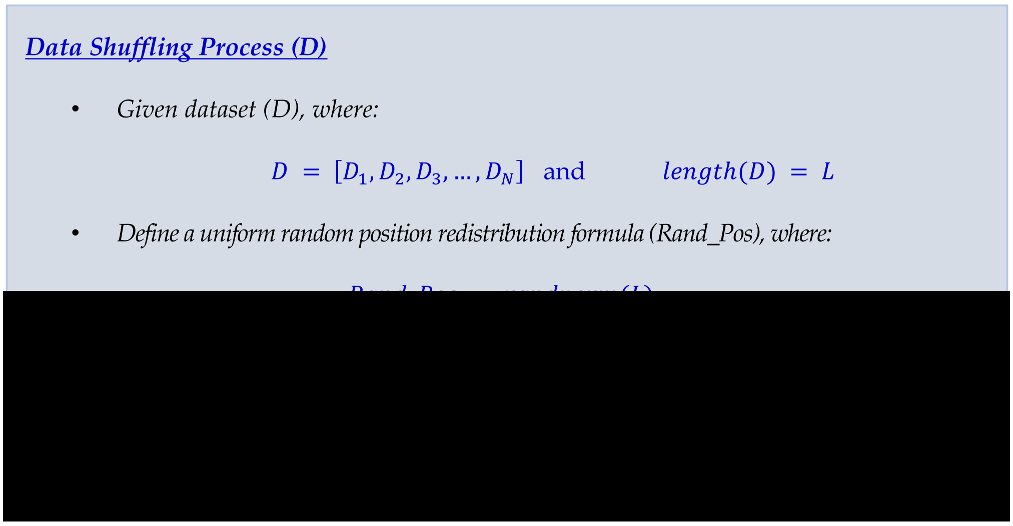
The data standardization process (DSP) is a fundamental step in the data preprocessing stage that is mainly used to provide a feature scaling to make sure features are on almost the same scale so that each feature is equally important. The DSP makes data easier to process by most ML algorithms [31]. In this research, we applied the process of standardization (Z-score normalization) where all features are rescaled to ensure the mean and the standard deviation are with a distribution value between 0 and 1, respectively. The Z-score normalization applied in this research is given by the following formula:

$$X_{stand} = \frac{X - mean\ (X)}{Standard\ Deviation\ (X)}$$

Z-score normalization is useful for several optimization algorithms such as gradient descent (GD), which is extensively employed by machine learning algorithms. Standardization aims to increase the accuracy of ML models and mitigate/avoid bias of ML classifiers.

3.1.5. Data Shuffling Process (DSH)

The data shuffling process (DSH) is a fundamental step in the data preprocessing stage that is mainly used for masking confidential numerical data of the target dataset prior to being trained/validated through machine learning models. Shuffling is basically redistributing the data points/records throughout several execution processes (or even through several computing machines). In this research, since we are about to train/test splitting, we performed a uniform shuffle at every epoch, which guarantees that every item has the same chance to occur at any position [32]. The data shuffling applied in this research is performed using the following procedure:



Data Shuffling Process (D)

- *Given dataset (D), where:*

$$D = [D_1, D_2, D_3, \ldots, D_N] \quad \text{and} \qquad length(D) = L$$

- *Define a uniform random position redistribution formula (Rand_Pos), where:*

[redacted]

3.1.6. Data Distribution Process (DDP)

The data distribution process (DDP) is an essential operation in every machine-learning-based project. The DDP is basically responsible for dividing the utilized dataset into different subsets to be used for machine learning model training and testing processes. In this research, to ensure optimum data splitting, we employed the random division approach (*dividerand*) to divide the targets into training, validation, and testing datasets using random indices. Specifically, the *dividerand* algorithm works as illustrated in Figure 3 below.

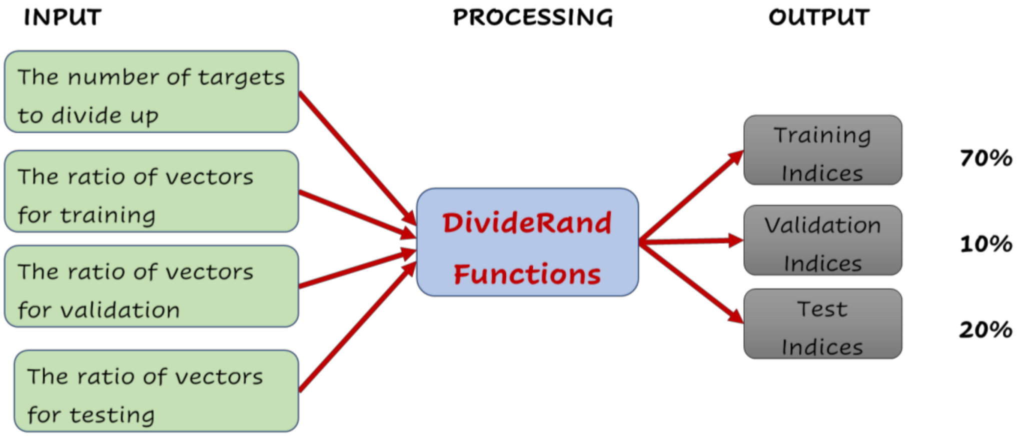

INPUT
PROCESSING
OUTPUT
The number of targets to divide up
The ratio of vectors for training
The ratio of vectors for validation
The ratio of vectors for testing
DivideRand Functions
Training Indices
Validation Indices
Test Indices
70%
10%
20%

Figure 3. The basic idea of random division of the dataset into three datasets.

In addition, to ensure optimum validation/testing process, we also applied k-fold cross-validation [33] during the learning process. In this approach, the dataset is arbitrarily split into k laminated folds, with each fold used as a test/validate dataset once while the other folds are consolidated together for use as a training dataset for the machine learning model generation. For every fold, the performance measures of the test dataset are computed and saved. Once all folds are processed, the overall performance measures are then calculated from all the measures stored for all k-folds. Figure 4 shows the schematic overview of k-fold cross-validation.

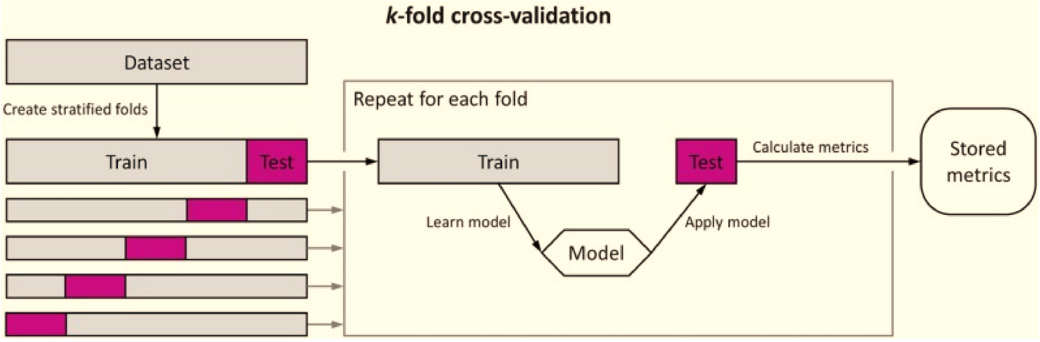


Figure 4. The basic idea of k-fold cross-validation for the employed dataset.

3.2. Implementation of the Learning Process (LP) Module

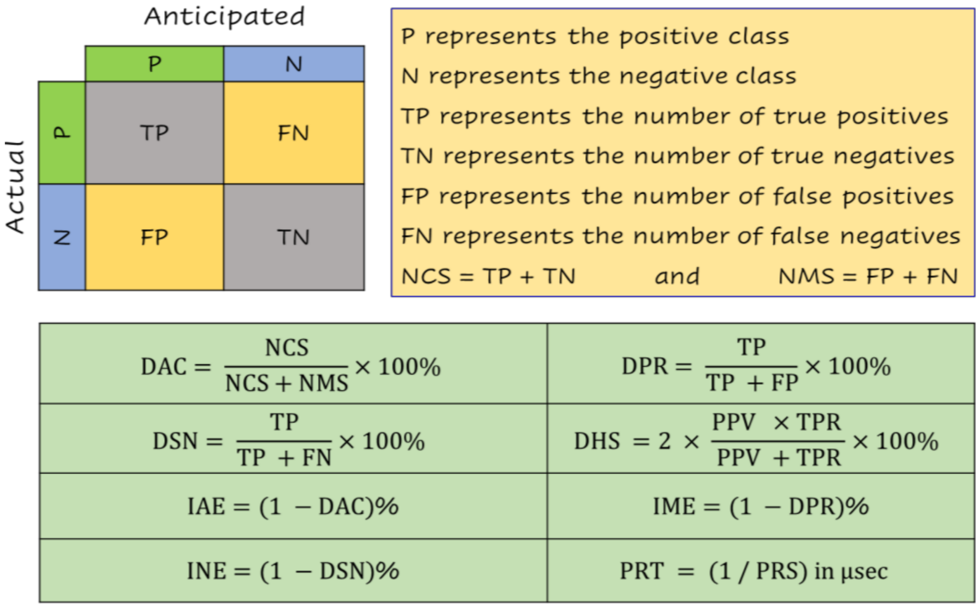
Hitherto, the implementation phases of the DP module have been investigated and analyzed. The DP module has distributed datasets that are ready for the learning process (learning includes training and testing), and thus, the next step is to process the preprocessed distributed datasets using an LP-module-based machine learning system. The main objective of this module is to train/test the developed IDS-based machine learning models along with their ensemble model, aiming to obtain the optimal performance trajectory recording maximum detection accuracy with the least detection error. In this research, the proposed solution approach evaluates the performance of three supervised machine learning methods (AdaBoosted decision tree [34], RUSBoosted decision tree [35], and bagged decision tree [36]) in order to provide more inclusive experiments and gain more insight into the solution approach. In addition, the ensemble learning model, making use of the three supervised machine learning methods, is developed to profile the behavioral features of IoT network traffic and identify the anomalous network traffic launched through the compromised IoT devices. We summarize specifications for the implemented machine learning models in Table 3.

Table 3. Specifications for the implemented machine learning models.

Classifier	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Model Preset	Decision trees	Decision trees	Decision trees	Ensemble learning
Learner Type	AdaBoosted	RUSBoosted	Bagged	Bagged, AdaBoosted, RUSBoosted
Maximum Number of Splits	20	20	20	1–611358
Number of Learners	30	30	30	10–500
Learning Rate	0.1	0.1	0.1	0.001–1
	Summary of General Implementation Specifications			
Feature Selection	Validation		Shuffling Process	
CCS Approach	5-fold cross-validation		Uniform shuffling at every epoch	
Data Distribution	Standardization Process		Computing Platform	
Divide-Rand	Z-score normalization		Windows 11/MATLAB2021/GPU+CPU	

3.3. Implementation of the Evaluation Process (EP) Module

The evaluation process (EP) is an essential activity to measure and regulate the quality metrics that check the compliance of the system with its requirements and objectives. To validate the efficiency of the ELBA-IoT system, we used the standard evaluation metrics [37] over the k-fold datasets to measure the performance of the ELBA-IoT system using the four aforementioned variants of machine learning models: the AdaBoosted decision tree (ABDT) model, RUSBoosted decision tree (RBDT) model, bagged decision tree (BGDT) model, and ensemble learning model (ELBA-IoT). Figure 5 recapitulates the overall system of measurements employed in this research to validate the system performance and compare the quality of several models [17].



P represents the positive class
N represents the negative class
TP represents the number of true positives
TN represents the number of true negatives
FP represents the number of false positives
FN represents the number of false negatives
NCS = TP + TN and NMS = FP + FN

$\text{DAC} = \frac{\text{NCS}}{\text{NCS} + \text{NMS}} \times 100\%$	$\text{DPR} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\%$
$\text{DSN} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\%$	$\text{DHS} = 2 \times \frac{\text{PPV} \times \text{TPR}}{\text{PPV} + \text{TPR}} \times 100\%$
$\text{IAE} = (1 - \text{DAC})\%$	$\text{IME} = (1 - \text{DPR})\%$
$\text{INE} = (1 - \text{DSN})\%$	$\text{PRT} = (1 / \text{PRS}) \text{ in } \mu\text{sec}$

Figure 5. System evaluation: summary of performance indication factors.

According to the figure (Figure 5), the performance of the developed models was evaluated in terms of confusion matrix analysis to discuss the number of positive and negative samples (predicted as true or false), the number of correctly classified samples (NCS#), the number of misclassified samples (NMS#), the number of correctly-classified samples (NCS#), the detection accuracy proportion (DAC%), the detection precision proportion (DPR%), the detection sensitivity proportion (DSN%), the detection harmonic score (DHS%), the detection inaccuracy error (IAE%), the detection imprecision error (IME%), and the detection insensitivity error (INE%). In addition, for IoT applications, the functional time is a crucial factor for such energy-aware devices; therefore, we evaluated the proposed system in terms of inference overhead, represented by two factors: (1) the prediction speed (PRS), which measures the number of samples predicted within a time unit (1 time unit = 1 s), and conversely, (2) the prediction time (overhead/PRT), which measures the amount of time needed to provide the detection for single-sample traffic.

4. Results and Discussion

ELBA-IoT is a defense system that can be used for botnet detection and classification. In this section, we provide the results obtained for the performance evaluation of ELBA-IoT and the other ML models at three levels of defense (using three classifiers): the binary classifier, which is used to identify the IoT traffic as either normal or anomaly (botnet); the ternary classifier, which is used to classify the IoT traffic into normal, Mirai botnet, or Bashlite (Gafgyt) botnet; and the multiclass classifier, which is used to classify the IoT traffic into normal, Mirai_Doorbell, Mirai_Thermostat, Mirai_Baby_Monitor, Mirai_Security_Camera, Mirai_Webcam, Gafgyt_Doorbell, Gafgyt_Thermostat, Gafgyt_Baby_Monitor, Gafgyt_Security_Camera, or Gafgyt_Webcam.

To begin, Table 4 presents the results of performance evaluation metrics obtained for the binary classifier (normal vs. anomaly) using the four aforementioned botnet detection models: Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT). According to the table, all binary-class models exhibited an outstanding detectability of botnets, scoring accuracy rates of 99.5–100%, with explicit superiority for ELBA-IoT, which was able to detect all IoT botnet traffic with a 0.0% inaccuracy error rate.

Table 4. Experimental outcomes for the performance of binary classifier using Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT).

	DAC	DPR	DSN	DHS	IAE	IME	INE	NMS#	NCS#
ADA-IoT	99.9%	99.7%	99.5%	99.6%	0.1%	0.3%	0.5%	611	610748
RUS-IoT	99.5%	99.3%	99.3%	99.3%	0.5%	0.7%	0.7%	3056	608303
BAG-IoT	99.7%	99.5%	99.4	99.4%	0.3%	0.5%	0.6%	1834	609525
ELBA-IoT	100%	100%	100%	100%	0.0%	0.0%	0.0%	13	611346

Moreover, Table 5 presents the results of performance evaluation metrics obtained for the ternary classifier (normal, Mirai, or Bashlite) using the four aforementioned botnet detection models: Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT). According to the table, all ternary-class models exhibited an outstanding detectability of botnets, scoring accuracy rates of 99.5–100%, with explicit superiority for ELBA-IoT, which was able to detect all IoT botnet traffic with a 0.0% inaccuracy error rate.

Table 5. Experimental outcomes for the performance of ternary classifier using Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT).

	DAC	DPR	DSN	DHS	IAE	IME	INE	NMS#	NCS#
ADA-IoT	99.8%	99.7%	99.7%	99.7%	0.1%	0.3%	0.3%	672	610687
RUS-IoT	99.4%	99.4%	99.2%	99.3%	0.6%	0.4%	0.8%	3655	607704
BAG-IoT	99.5%	99.5%	99.4%	99.4%	0.5%	0.5%	0.6%	3117	608242
ELBA-IoT	100%	100%	100%	100%	0.0%	0.0%	0.0%	30	611329

Moreover, Table 6 presents the results of performance evaluation metrics obtained for the multiclassifier (normal, Mirai_Doorbell, Mirai_Thermostat, Mirai_Baby_Monitor, Mirai_Security_Camera, Mirai_Webcam, Gafgyt_Doorbell, Gafgyt_Thermostat, Gafgyt_Baby_Monitor, or Gafgyt_Security_Camera) using the four aforementioned botnet detection models: Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT). According to the table, all multiclass models exhibited an outstanding detectability of botnets, scoring accuracy rates of 96.2–99.6%, with explicit superiority for ELBA-IoT, which was able to detect all IoT botnet traffic with a 0.4% inaccuracy error rate.

Table 6. Experimental outcomes for the performance of multiclassifier using Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT).

	DAC	DPR	DSN	DHS	IAE	IME	INE	NMS#	NCS#
ADA-IoT	97.3%	95.5%	92.2%	93.9%	2.7%	4.5%	7.8%	16506	594853
RUS-IoT	97.7%	96.9%	95.7%	96.3%	2.3%	3.1%	4.3%	14061	597298
BAG-IoT	96.2%	92.4%	90.6%	91.5%	3.8%	7.6%	9.4%	23231	588128
ELBA-IoT	99.6%	98.4%	97.1%	97.7%	0.4%	1.6%	2.9%	2445	608914

In addition, due to the importance of ELBA-IoT for the multiclassifier, we traced its performance trajectory in terms of minimum classification error (MSE) vs. training iterations. Therefore, Figure 6, below, illustrates the performance progression for ELBA-IoT with the multiclassifier by tracing the MSE factor during 30 iterations. To optimize the performance of the ensemble learning model, the Bayesian optimizer was used with acquisition function based on the expected improvement per second plus. Moreover, the ensemble learner employed three powerful ensemble methods (RUSBoosted, AdaBoosted, and bagged), utilizing 10 learners with maximum splits number of 15 splits, number of predictors to sample of 1–116, and optimized learning rate of 0.192. Accordingly, the best point minimum error was recorded after 11 iterations; the classifier recorded the minimum classification error of 3.8 × 10^{-3} MSE (~3.8%) before becoming fixed and stable over the remaining iterations.

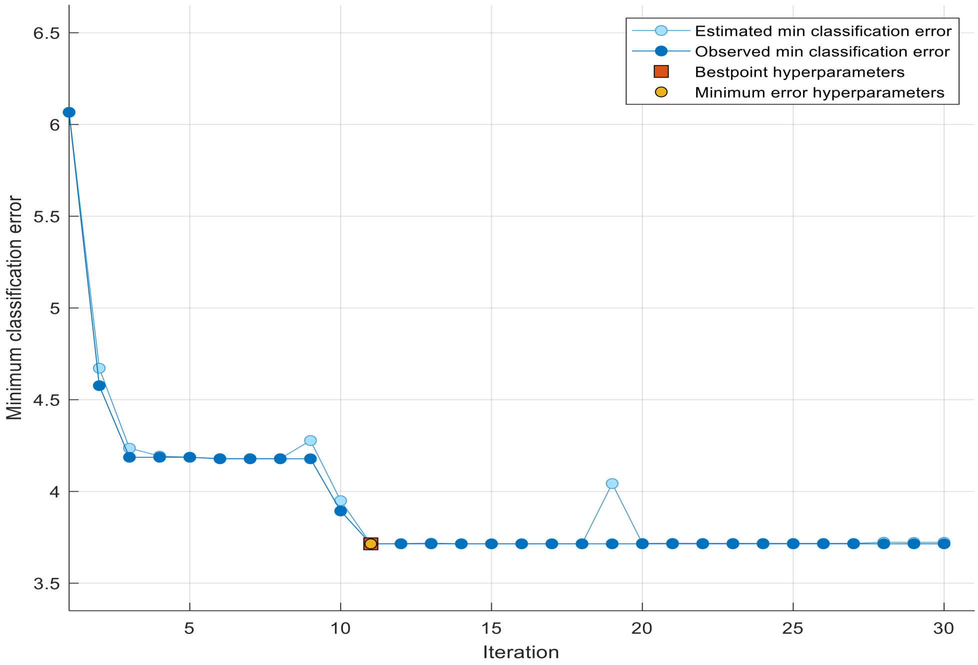


Figure 6. Performance progression of ELBA-IoT optimized using 30 iterations.

Furthermore, to gain more insight into the inference overhead, Table 7 provides the time complexity results for the developed multiclass inference systems characterized into two aspects: the prediction speed (PRS) and the prediction time (overhead/PRT) for single-sample traffic. According to the table, ELBA-IoT recorded the maximum prediction speed and, thus, it has the least inference overhead, recording a prediction time of 40 µ-Second. Such a reasonable amount of time (at the micro scale) permits the ELBA-IoT model to be efficiently applied at the IoT systems (as gateway devices) to provide botnet classification for IoT traffic at earliest time and low overhead.

Table 7. Experimental outcomes for the performance of multiclassifier using Model 1 (ADA-IoT), Model 2 (RUS-IoT), Model 3 (BAG-IoT), and Model 4 (ELBA-IoT).

	ADA-IoT	RUS-IoT	BAG-IoT	ELBA-IoT
PRS (Samples/Sec)	12,000	13,000	12,000	25,000
PRT (in µ-Second)	83.33	76.92	83.33	40.00

Finally, Table 8 presents the comparative analysis of our ELBA-IoT system with other present state-of-art IoT-based botnet detection schemes utilizing machine/deep learning approaches in the same area of study. The table compares our best empirical results recorded for ELBA-IoT with the respective factors reported in existing studies. The reported comparison metrics include the detection model (learning model), the attack categories involved in the detection system, the number of output classes of each detection model, and the validation accuracy proportion for the detection models.

Table 8. Comparison with other existing ML-based IoT-IDS Systems.

Paper/ Year	Detection Model	Attack Categories	Number of Classes	Validation Accuracy
[38]/2017	HAEs	DOS, PROBE, R2L, U2R	5 Classes	88.65%
[27]/2018	BLSTM-RNN	MIRAI, UDP, ACK, DNS	5 Classes	97.5%
[26]/2018	DG-CNN	PORT SCANNING, MIRAI, QBOT, DICTIONARY	2 Classes	92.0%
[39]/2019	kNN	TELNET, HTTP_POST, HTTP_GET	3 Classes	94.45%
[40]/2019	SVM	DOS, PROBE, R2L, U2R	5 Classes	81.00%
[41]/2019	Hybrid-ML	DOS, PROBE, R2L, U2R	5 Classes	85.20%
[25]/2020	LSTM + RNN	MIRAI AND ITS VARIANTS	2 Classes	99.30%
[42]/2020	S-CNN	DOS, PROBE, R2L, U2R	5 Classes	98.20%
[43]/2020	D-CNN	MIRAI HAJIME, BRICKERBOT, MASUTA, SORA	4 Classes	90.00%
[44]/2021	SL-BMM-CE	MIRAI + BASHLITE	7 Classes	99.20%
[45]/2022	ADA-DT	DDoS, INJC, MITM, PSWD, SCAN, XSS, BKDR, RNSM	10 Classes	98.60%
ELBA-IoT	EL-DTs	MIRAI + BASHLITE	10 Classes	99.60%

Consequently, the comparison table (Table 8) contemplates 11 different intrusion/botnet detection models for the IoT environment, spanning from 2017 to 2022, in addition to our proposed ELBA-IoT, which is based on ensemble learning of decision trees (EL-DTs). The considered models include HAEs-2017 (hybrid autoencoders model, comprising autoencoders and denoising autoencoder) [38], BLSTM-RNN-2108 (cascade model, comprising bidirectional long short-term memory and recurrent neural networks) [27], DG-CNN-2018 (deep graph convolutional neural networks model) [26], kNN-2019 (k-Nearest-Neighbor-based IDS model) [39], SVM-2019 (support-vector-machine-based IDS model) [40], Hybrid-ML-2019 (hybrid machine learning model composed of the decision tree, random forest, kNN, and deep neural networks) [41], LSTM-RNN-2020 (long short-term memory and recurrent-neural-network-based IDS model) [25], S-CNN-2020 (shallow-convolutional-neural-network-based IDS model) [42], D-CNN-2020 (deep-convolutional-neural-network-based IDS model) [43], SL-BMM-CE-2021 (statistical-learning-based beta mixture model (BMM) and correntropy-based IDS model) [44], and ADA-DT-2022 (AdaBoosted-decision-tree-based IDS model) [45]. Based on the comparisons provided in the table, we can undoubtedly conclude that our ELBA-IoT model is conspicuously superior, as it provides the top detection accuracy performance for a multiclassifier with 10 labeled classes at the output layer with low inference overhead (40 µ-seconds) that can be adapted effectively with the time-aware and low-power devices of the IoT system. In addition, our ELBA-IoT improved the accuracy of identification of attacks by 1% over the 10-class classifier, 1.4–18.6% over the 5-class classifiers, and 0.3–7.6% over other lower-class classifiers (2-, 3-, and 4-class) for the identified IoT-IDS models in Table 8.

5. Conclusions and Future Directions

An autonomous, lightweight, intelligent, and accurate intrusion detection system for IoT networks—called ELBA-IoT—was proposed, developed, and evaluated in this paper. ELBA-IoT makes use of supervised ensemble learning methods to profile the behavioral features of IoT network traffic and identify the anomalous network traffic launched through compromised IoT devices. In addition, the proposed solution approach evaluated the performance of three supervised machine learning models (AdaBoosted, RUSBoosted, and bagged) in order to provide more inclusive experiments and gain more insight into the solution approach. Moreover, the N-BaIoT-2021 dataset—a comprehensive and contemporary dataset comprising real-world IoT network traffic—was used to evaluate the performance of ELBA-IoT. Eventually, the empirical outcomes demonstrated the superiority of ELBA-IoT over other existing solutions, scoring a high detection accuracy (99.6%) with low inference overhead (40 µ-seconds). As for future work, one recommended extension of this work is to deploy ELBA-IoT using physical IoT gateway devices to provide real-time botnet detection facilities for IoT networks (this may include deploying IoT-enabled devices, such as the

IoT-based Raspberry Pi, ARM Cortex, or Arduino). Such deployment can be accompanied by further investigation on the different low-power IoT nodes' concerns, such as energy consumption, memory usage, and communication complexity. We may also aim to produce a new dataset for IoT botnets using this investigation at several peak times and several IoT applications of interest.

Author Contributions: Conceptualization, Q.A.A.-H.; methodology, Q.A.A.-H.; software, Q.A.A.-H.; validation, Q.A.A.-H. and M.A.-D.; formal analysis, Q.A.A.-H. and M.A.-D.; investigation, Q.A.A.-H.; resources, Q.A.A.-H. and M.A.-D.; data curation, Q.A.A.-H.; writing—original draft preparation, Q.A.A.-H. and M.A.-D.; writing—review and editing, Q.A.A.-H. and M.A.-D.; visualization, Q.A.A.-H.; supervision, Q.A.A.-H.; project administration, Q.A.A.-H.; funding acquisition, Q.A.A.-H. and M.A.-D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

References

1. Albulayhi, K.; Smadi, A.A.; Sheldon, F.T.; Abercrombie, R.K. IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses. *Sensors* **2021**, *21*, 6432. [CrossRef] [PubMed]
2. Statistical Portal. Internet of Things (IoT) Connected Devices Installed Base Worldwide from 2015 to 2025 (in Billions). Available online: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (accessed on 24 December 2018).
3. Rose, K.; Eldridge, S.; Chapin, L. The Internet of Things: An Overview Understanding the Issues and Challenges of a More Connected World. 2015. Available online: http://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151022.pdf (accessed on 20 January 2022).
4. Dambaye, S.S.; Kolhe, M.V.L. A Survey: Managing Resource-Constrained Devices in IoT. *Int. J. Innov. Res. Comput. Commun. Eng.* **2016**, *4*, 21011–21015.
5. Al-Haija, Q.A. On the Security of Cyber-Physical Systems Against Stochastic Cyber-Attacks Models. In Proceedings of the 2021 IEEE International IoT, Electronics, and Mechatronics Conference (IEMTRONICS), Toronto, ON, Canada, 21–24 April 2021; pp. 1–6. [CrossRef]
6. Al Dalaien, M.N.; Bensefia, A.; Hoshang, S.A.; Bathaqili, A.R.A.; Xu, X.; Mohanan, V.; Budiarto, R.; Aldmour, I. Internet of Things (IoT) Security and Privacy. In *Powering the Internet of Things with 5G Networks*; Mohanan, V., Budiarto, R., Aldmour, I., Eds.; IGI Global: Hershey, PA, USA, 2018; pp. 247–267. [CrossRef]
7. Albulayhi, K.; Sheldon, F.T. An Adaptive Deep-Ensemble Anomaly-Based Intrusion Detection System for the Internet of Things. In Proceedings of the 2021 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 10–13 May 2021; pp. 187–196. [CrossRef]
8. Sagi, O.; Rokach, L. Ensemble learning: A survey. *WIREs Data Min. Knowl. Discov.* **2018**, *8*, e1249. [CrossRef]
9. Tsogbaatar, E.; Bhuyan, M.H.; Taenaka, Y.; Fall, D.; Gonchigsumlaa, K.; Elmroth, E.; Kadobayashi, Y. SDN-enabled IoT anomaly detection using ensemble learning. In Proceedings of the 16th International Conference on Artificial Intelligence Applications and Innovations (AIAI), Halkidiki, Greece, 5–7 June 2020.
10. Median, Y.; Bogadana, 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]
11. Basavaraj, D.; Tayeb, S. Towards a Lightweight Intrusion Detection Framework for In-Vehicle Networks. *J. Sens. Actuator Netw.* **2022**, *11*, 6. [CrossRef]
12. Samara, M.A.; Bennis, I.; Abouaissa, A.; Lorenz, P. A Survey of Outlier Detection Techniques in IoT: Review and Classification. *J. Sens. Actuator Netw.* **2022**, *11*, 4. [CrossRef]
13. Alrubayyi, H.; Goteng, G.; Jaber, M.; Kelly, J. Challenges of Malware Detection in the IoT and a Review of Artificial Immune System Approaches. *J. Sens. Actuator Netw.* **2021**, *10*, 61. [CrossRef]
14. Ioannou, C.; Vassiliou, V. Network Attack Classification in IoT Using Support Vector Machines. *J. Sens. Actuator Netw.* **2021**, *10*, 58. [CrossRef]
15. Ramadan, R.A. Efficient Intrusion Detection Algorithms for Smart Cities-Based Wireless Sensing Technologies. *J. Sens. Actuator Netw.* **2020**, *9*, 39. [CrossRef]
16. Abu Al-Haija, Q. Top-Down Machine Learning-Based Architecture for Cyberattacks Identification and Classification in IoT Communication Networks. *Front. Big Data* **2022**, *4*, 782902. [CrossRef]
17. Abu Al-Haija, Q.; Al-Badawi, A. Attack-Aware IoT Network Traffic Routing Leveraging Ensemble Learning. *Sensors* **2022**, *22*, 241. [CrossRef] [PubMed]
18. Al-Haija, Q.A.; Saleh, E.; Alnabhan, M. Detecting Port Scan Attacks Using Logistic Regression. In Proceedings of the 2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Khobar, Saudi Arabia, 6–8 December 2021; pp. 1–5. [CrossRef]

19. Tsogbaatar, E.; Bhuyan, M.H.; Taenaka, Y.; Fall, D.; Gonchigsumlaa, K.; Elmroth, E.; Kadobayashi, Y. DeL-IoT: A deep ensemble learning approach to uncover anomalies in IoT. *Internet Things* **2021**, *14*. [CrossRef]
20. Rezaei, A. Using Ensemble Learning Technique for Detecting Botnet on IoT. *SN Comput. Sci.* **2021**, *4*, 148. [CrossRef]
21. Özçelik, M.; Chalabianloo, N.; Gür, G. Software-Defined Edge Defense against IoT-Based DDoS. In Proceedings of the 2017 IEEE International Conference on Computer and Information Technology (CIT 17), Helsinki, Finland, 21–23 August 2017. [CrossRef]
22. 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 15), Mamkomg, China, 14–16 December 2015. [CrossRef]
23. Yang, L.; Shami, A. A Lightweight Concept Drift Detection and Adaptation Framework for IoT Data Streams. *IEEE Internet Things Mag.* **2021**, *4*, 96–101. [CrossRef]
24. Qaddoura, R.; Al-Zoubi, A.M.; Almomani, I.; Faris, H. A Multi-Stage Classification Approach for IoT Intrusion Detection Based on Clustering with Oversampling. *Appl. Sci.* **2021**, *11*, 3022. [CrossRef]
25. Shi, W.C.; Sun, H.M. DeepBot: A time-based botnet detection with deep learning. *Soft. Comput.* **2020**, *24*, 16605–16616. [CrossRef]
26. Nguyen, H.-T.; Ngo, Q.-D.; Le, V.-H. IoT Botnet Detection Approach Based on PSI graph and DGCNN classifier. In Proceedings of the 2018 IEEE International Conference on Information Communication and Signal Processing (ICICSP), Singapore, 28–30 September 2018; pp. 118–122. [CrossRef]
27. McDermott, C.D.; Majdani, F.; Petrovski, A.V. Botnet Detection in the Internet of Things using Deep Learning Approaches. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
28. Stiawan, D.; Suryani, M.E.; Susanto; Idris, M.Y.; Aldalaien, M.N.; Alsharif, N.; Budiarto, R. Ping Flood Attack Pattern Recognition Using a K-Means Algorithm in an Internet of Things (IoT) Network. *IEEE Access* **2021**, *9*, 116475–116484. [CrossRef]
29. Al-Haija, Q.A.; Smadi, A.A.; Allehyani, M.F. Meticulously Intelligent Identification System for Smart Grid Network Stability to Optimize Risk Management. *Energies* **2021**, *14*, 6935. [CrossRef]
30. Chandra, B.E.; Karthikeyan, E. Sigmis: A feature selection algorithm using the correlation-based method. *J. Algorithms Comput. Technol.* **2012**, *6*, 385–394.
31. Singh, D.; Birmohan, S. Investigating the impact of data normalization on classification performance. *Appl. Soft Comput.* **2020**, *97*, 105524. [CrossRef]
32. Al-Haija, Q.A.; Alsulami, A.A. High Performance Classification Model to Identify Ransomware Payments for Heterogeneous Bitcoin Networks. *Electronics* **2021**, *10*, 2113. [CrossRef]
33. Abu Al-Haija, Q.; Krichen, M.; Abu Elhaija, W. Machine-Learning-Based Darknet Traffic Detection System for IoT Applications. *Electronics* **2022**, *11*, 556. [CrossRef]
34. Stamp, M. A survey of machine learning algorithms and their application in information security. In *Guide to Vulnerability Analysis for Computer Networks and Systems*; Springer: Cham, Switzerland, 2018; pp. 33–55.
35. Timčenko, V.; Gajin, S. Ensemble classifiers for supervised anomaly-based network intrusion detection. In Proceedings of the 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 7–9 September 2017; pp. 13–19. [CrossRef]
36. Gaikwad, D.P.; Thool, R.C. Intrusion detection system using bagging with partial decision treebase classifier. *Procedia Comput. Sci.* **2015**, *49*, 92–98. [CrossRef]
37. Al-Haija, Q.A.; Ishtaiwi, A. Multiclass Classification of Firewall Log Files Using Shallow Neural Network for Network Security Applications. In *Soft Computing for Security Applications. Advances in Intelligent Systems and Computing*; Ranganathan, G., Fernando, X., Shi, F., El-Allioui, Y., Eds.; Springer: Singapore, 2022; Volume 1397. [CrossRef]
38. Aygun, R.C.; Yavuz, A.G. Network anomaly detection with stochastically improved autoencoder based models. In Proceedings of the 4th International Conference on Cyber Security and Cloud Computing, New York, NY, USA, 26–28 June 2017; pp. 193–198.
39. Kumar, A.; Lim, T.J. EDIMA: Early detection of IoT malware network activity using machine learning techniques. In Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Limerick, Ireland, 15–18 April 2019; pp. 289–294.
40. Ioannou, C.; Vassiliou, V. Classifying Security Attacks in IoT Networks Using Supervised Learning. In Proceedings of the 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini Island, Greece, 29–31 May 2019; pp. 652–658.
41. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access* **2019**, *7*, 82512–82521. [CrossRef]
42. Abu Al-Haija, Q.; Sabatto, S.Z. An Efficient Deep-Learning-Based Detection and Classification System for Cyber-Attacks in IoT Communication Networks. *Electronics* **2020**, *9*, 2152. [CrossRef]
43. Jung, W.; Zhao, H.; Sun, M.; Zhou, G. IoT botnet detection via power consumption modeling. *Smart Health* **2020**, *15*, 100103. [CrossRef]
44. Ashraf, J.; Keshk, M.; Moustafa, N.; Abdel-Basset, M.; Khurshid, H.; Bakhshi, A.D.; Mostafa, R.R. IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities. *Sustain. Cities Soc.* **2021**, *72*, 103041. [CrossRef]
45. Abu Al-Haija, Q.; al Badawi, A.; Bojja, G.R. Boost-Defence for resilient IoT networks: A head-to-toe approach. *Expert Syst.* **2022**, *39*, e12934. [CrossRef]