Journal of
Sensor and Actuator Networks

Article

A Novel Multi Algorithm Approach to Identify Network Anomalies in the IoT Using Fog Computing and a Model to Distinguish between IoT and Non-IoT Devices

Rami J. Alzahrani [1,2,*] and Ahmed Alzahrani [1]

1 Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia
2 Department of Computer Science, Faculty of Computer Science and Information Technology, Al-Baha University, Al-Baha 65799, Saudi Arabia
* Correspondence: ralzahrani0654@stu.kau.edu.sa

Abstract: Botnet attacks, such as DDoS, are one of the most common types of attacks in IoT networks. A botnet is a collection of cooperated computing machines or Internet of Things gadgets that criminal users manage remotely. Several strategies have been developed to reduce anomalies in IoT networks, such as DDoS. To increase the accuracy of the anomaly mitigation system and lower the false positive rate (FPR), some schemes use statistical or machine learning methodologies in the anomaly-based intrusion detection system (IDS) to mitigate an attack. Despite the proposed anomaly mitigation techniques, the mitigation of DDoS attacks in IoT networks remains a concern. Because of the similarity between DDoS and normal network flows, leading to problems such as a high FPR, low accuracy, and a low detection rate, the majority of anomaly mitigation methods fail. Furthermore, the limited resources in IoT devices make it difficult to implement anomaly mitigation techniques. In this paper, an efficient anomaly mitigation system has been developed for the IoT network through the design and implementation of a DDoS attack detection system that uses a statistical method that combines three algorithms: exponentially weighted moving average (EWMA), K-nearest neighbors (KNN), and the cumulative sum algorithm (CUSUM). The integration of fog computing with the Internet of Things has created an effective framework for implementing an anomaly mitigation strategy to address security issues such as botnet threats. The proposed module was evaluated using the Bot-IoT dataset. From the results, we conclude that our model has achieved a high accuracy (99.00%) with a low false positive rate (FPR). We have also achieved good results in distinguishing between IoT and non-IoT devices, which will help networking teams make the distinction as well.

Keywords: anomaly mitigation; internet of things (IoT); intrusion detection system (IDS); fog computing; classification algorithms; machine learning; cybersecurity

check for updates

Citation: Alzahrani, R.J.; Alzahrani, A. A Novel Multi Algorithm Approach to Identify Network Anomalies in the IoT Using Fog Computing and a Model to Distinguish between IoT and Non-IoT Devices. *J. Sens. Actuator Netw.* **2023**, *12*, 19. https://doi.org/10.3390/jsan12020019

Academic Editors: Subhas Mukhopadhyay, Yoshiyasu Takefuji and Enrico Vezzetti

Received: 10 February 2023
Revised: 24 February 2023
Accepted: 27 February 2023
Published: 28 February 2023

1. Introduction

A product of the digital revolution is the Internet of Things (IoT), a modern technology that connects various devices, for example, smart TVs, printers, cameras, smartphones, and smartwatches, via cyberspace. The IoT introduces new services and apps to numerous users and improves their quality of life. The majority of IoT things and items do not need an extensive capacity. In addition, the total of storage and handing out capacity available in the IoT is limited. As a result, a huge number of sensitive data are stored in clouds. This reduces the costs and effort, while increasing the availability and accessibility of provided services. However, the fact that these IoT apps and services can be accessed "anytime and anywhere," is a key challenge for the security and privacy of the data stored on these apps and services, in addition to other IoT-related challenges, such as performance and cost, the management of the heterogeneous environment (hardware conflict/data conflict), resource constraints (computational, power, and storage capacity), accessibility, and scalability.

There is no standard structure or protocol (transferring, processing, storing, etc.) when one is dealing with a large amount of data (big data). The expansion of the IoT has led to an increase in various challenges [1]. A number of these concerns frequently manifest as network anomalies, such as a diversion from normal network activity. Nowadays, there are an increasing number of IoT devices, but there are some limitations of the cloud [2]:

- Huge network utilization: because network bandwidth is an unusual resource that will not be able to keep up with the increasing amount of data generated by IoT devices, network usage will rise dramatically.
- High energy consumption: as the number of servers in cloud data centers grows, so does power consumption in the data centers, raising the cost of energy consumption.
- Large latency: as the number of IoT devices and applications grows, the traditional cloud will be unable to deliver the low latency needed by delay-sensitive or real-time applications, negatively impacting performance levels.
- Reliance: Everything today is dependent on a connection to the Internet. We need an Internet connection to access cloud services. Internet outages can lead to dangerous situation where time-sensitive applications are concerned.
- Non-essential data storage: Several apps are limited by the computer and storage capacity, severely limiting their ability to store, transmit, and process data. The storage of unnecessary data in the cloud can result in high maintenance costs and inoperative data analytics.
- Mobility and restricted control: customers of cloud computing have minimal control over their applications, data, and services, which is a major issue.
- Privacy and security concerns: There have been numerous security breaches involving sensitive data in the cloud. Fog computing, on the other hand, improves the security approach by offering a higher level of security and establishing numerous walls. Implementing machine learning in fog computing can improve the network routing performance, as well as provide better insights and solutions.

In cloud computing, fog computing is a defensive strategy against the ever-expanding security threats, principally DDoS attacks [3]. Fog acts by means of a clarifying layer for the generated traffic and is located between the user and the cloud. In [4], the authors worked to improve the overall network performance and reduce the traffic transferred to the cloud by offering services toward the network's edge. Fog computing improves the network's capability.

1.1. Research Motivation

Several strategies have been developed to reduce the anomalies in IoT networks, such as DDoS. To improve the accuracy of the anomaly detection rate and lower the false positive rate (FPR), some schemes use statistical or machine learning methodologies in anomaly-based IDS to mitigate an attack. Despite the proposed anomaly mitigation techniques, the mitigation of DDoS attacks on IoT networks remains a concern. Because of the similarity between DDoS and a normal network, leading to problems such as a high FPR, low accuracy, and a low detection rate, the majority of anomaly mitigation methods have failed. Furthermore, the limited resources in IoT devices have made the implementation of anomaly mitigation techniques more difficult. Numerous solutions have been presented to mitigate anomalies in IoT networks on the victim side. However, the mitigation of attacks such as DDoS still poses difficulties due to the similarity of the DDoS traffic flow to the normal traffic flow, and the mitigation system in most of the studies addresses the victim side. A perfect DDoS detection system will detect DDoS attack traffic in real time and mitigate the attack as near the source as possible in order to prevent additional network destruction. This leads to two main objectives: high accuracy and early detection. Accuracy refers to a system's ability to detect DDoS attacks with a high true positive rate and a low FPR, while early detection refers to detecting a DDoS attack as close to the origin of the attack as possible, for example, at the local switch/router nearest to the attack origin. The primary issue with existing DDoS attack detection approaches is their inability to accurately and quickly detect both high- and low-intensity DDoS attacks in large-scale networks because of shortcomings in their performance, expandability, and information exchange abilities. Therefore, it is critical to develop new techniques for the accurate and timely detection of both high- and low-intensity DDoS attacks. In this research, our proposed system is placed at the fog level to monitor the outgoing traffic and detect the attack at an early stage. One important characteristic of our system is the ability to treat the IoT traffic in a different way since we are concerned about the attacks generated by IoT devices.

1.2. Contributions

We have developed an efficient anomaly mitigation system for the IoT network by designing and implementing a DDoS attack detection system relying on a statistical method that combines three algorithms: exponentially weighted moving average (EWMA), the cumulative sum algorithm (CUSUM), and K-nearest neighbors (KNN). The contributions of this study can be considered as critical in the field of IoT devices, where the deployment of anomaly mitigation strategies is challenging because of limited resources. Within the general framework of the suggested approach, several key contributions of this study stand out.

- This novel idea of combining the three algorithms in one anomaly-based IDS module to mitigate the DDoS attack in the fog layer in the IoT ensures accurate attack detection.
- Anomaly mitigation in the IoT achieves good performance in terms of a high detection rate and accurate classification of the network flow as normal or abnormal.
- The system distinguishes between IoT devices and non-IoT devices effectively.

1.3. Organization

The remainder of this paper is organized as follows. Section 2 presents the related works on IDS and statistical techniques in the IoT environment. Section 3 presents the methodology used in this paper. It describes the dataset, the methods of feature selection, dataset pre-processing, and the mathematical model. In Section 4, the approach is evaluated in terms of the experimental details, evaluation metrics, and experimental results and compared with other approaches. Section 5 provides the conclusion of the research paper and indicates the direction of future work.

2. Related Works

Numerous studies on the intrusion detection system (IDS) of DDoS attacks are presented in [5]. The authors implement different machine learning (ML) algorithms to analyze their performance in detecting DDoS attacks [5]. The researchers in [6] categorize the traffic in the IoT networks. They provide a comprehensive review of current and previous studies on IoT traffic characterization in relation to the IoT apps and their designs [6]. Several strategies are developed to reduce the anomalies in IoT networks, such as DDoS. To increase the accuracy of the anomaly mitigation system and lower the FPR, some schemes use statistical or machine learning methodologies in the anomaly-based intrusion detection systems (IDS) to mitigate an attack. Despite the proposed irregularity reduction techniques, in IoT traffic, the mitigation of DDoS attacks remains a concern. Because of the similarity between DDoS and normal network flows, leading to problems such as a high FPR, low accuracy, and a low detection rate (DR), the majority of anomaly mitigation methods fail. Furthermore, the limited resources in IoT devices obstruct the implementation of anomaly mitigation techniques. In the literature, we have classified the network anomaly mitigation schemes in the IoT into three parts, as shown in Figure 1. In this research, we have built our model based on anomaly-based detection techniques using both statistical techniques and machine learning.

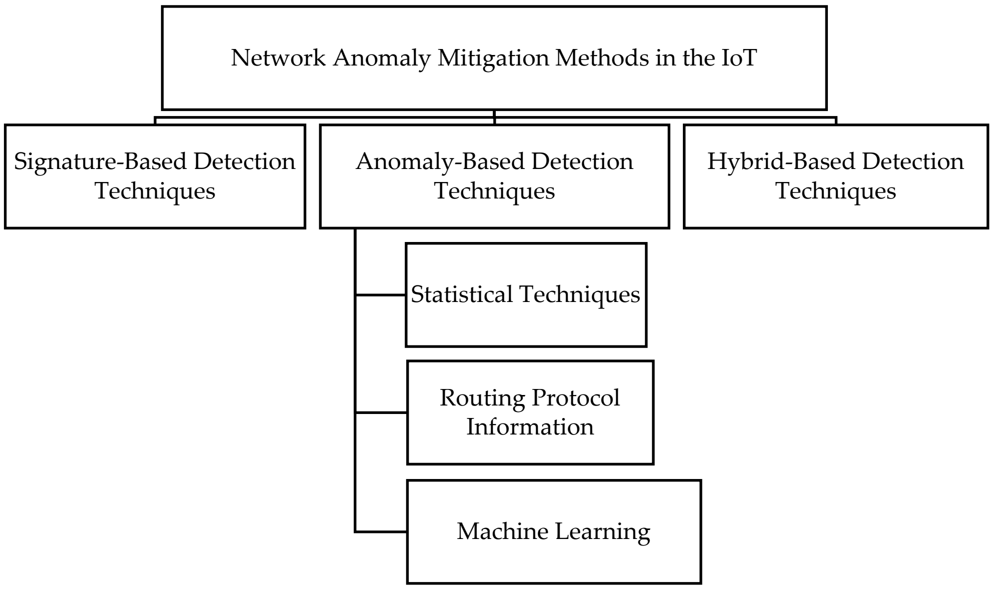

Network Anomaly Mitigation Methods in the IoT
Signature-Based Detection Techniques
Anomaly-Based Detection Techniques
Hybrid-Based Detection Techniques
Statistical Techniques
Routing Protocol Information
Machine Learning

Figure 1. The classification of network anomaly mitigation methods in the IoT.

In [7], CUSUM-Entropy is used to examine packet header fields to improve accuracy. By examining these fields, it is possible to distinguish between normal and non-normal traffic, achieving a high DR and a low FPR.

In [8], the authors use the IDS CUSUM solution. In this study, there is an example in which both DDoS and reflective DDoS successfully detect using the CUSUM algorithm. This is why we have used this algorithm in the solution presented in our thesis.

In [9], CUSUM Flood is used to detect TCP SYN flooding attacks, which are amongst the most widely known DDoS attacks. A non-parametric implementation, the CUSUM algorithm, is used to accomplish a high DR with a low FPR. This is one of the reasons we have used CUSUM in our research.

In [10], the authors employ an entropy and hybrid approach to prevent attackers from training models and convincing them that the attack traffic is normal. Various fields from network traffic are observed to achieve the best possible result. However, in [11], the authors explain why statistical process control is implemented in the IDS, in which principal component analysis is used to solve the issues caused by a large amount of quality characteristics.

In [12], as per the authors, when it comes to the reliability and security of IoT networks, anomaly detection is the most important component. Due to their lack of security, IoT networks are extremely vulnerable to attacks.

ProfilIoT, the method proposed in [13], focuses on an individual port by implementing an ML-based technique to classify IoT devices in the network. The objective is to determine whether the traffic stream is associated with any specifically identified IoT device. The proposed method can distinguish IoT devices from non-IoT devices. Similarly, by using a network behavior-based technique, it can identify the model of a device in a network, such as an IP camera or a smart TV. A set of categorizations based on ML algorithms is executed in a multistage procedure using traffic movements generated by a specific device that could be identified by its IP address.

Similarly, in [14], the authors propose a method for recognizing unidentified IoT devices and distinguishing between IoT and non-IoT devices. In this method, the generated traffic is used to model the behavior of a device. The method employs a deep long short-term memory (LSTM) auto encoder network to display the TCP structures of each device and group the structures based on the Bayesian Hyper Parameter Tuning model. The proposed method appears to have high precision in matching devices to their characterized equivalents.

In the same way, the authors in [15] create an ML framework for categorizing IoT devices. A characterization procedure is developed based on a variety of traffic structures, including signaling patterns, action cycles, transmission protocols, and cipher suite algorithms. Furthermore, a framework employs a classification based on ML in a multistage situation that accurately identifies IoT gadgets. The proposed technique discusses the trade-offs between the speed, costs, and performance.

Similarly, to recognize new and unknown devices in the network, the research in [16] executes a mechanized categorization technique that uses the structures of the traffic generated by IoT devices. This technique defines the specifications of various devices by using the rich data transported through IoT network traffic streams. An actual IoT dataset is used to evaluate the effectiveness of the implemented technique. According to the evaluation findings, the implemented technique shows a satisfactory level of performance in identifying devices.

The DeL-IoT in [17] uses a deep ensembling learning algorithm to detect anomalous IoT activities (DEL). This system is designed to address data heterogeneity and data imbalance issues in the IoT environment, while also predicting anomalous activities. This is accomplished by incorporating software-defined networking into the IoT to handle various network features. The presented system predicts whether the device status is normal or abnormal.

In [18], the authors examine the various DoS attacks on a network. The prediction depends on the attention to defense concepts against various DoS attacks. The designed model integrates the victim-based DoS defense strategy with the transmission of packets via attack mitigation. The funnel procedures are assessed on the destination node so as to minimize overcrowding at the access link in the node. The suggested model incorporates node collaboration with expenses and latency in the load. The DoS traffic is approximated using the funneling model for incorporating statistical filtering against DDoS attacks.

In [19], the authors focus on IoT NIDS implemented by using ML because learning algorithms have a high rate of success in terms of confidentiality and security. The research study thoroughly examines NIDSs implementing various characteristics of learning methods for the IoT.

In [20], the authors use a multilayer perceptron (MLP) model to present a lightweight IDS depending on a vector space representation. They compare the proposed IDS to the Australian Defense Force Academy Linux Dataset (ADFA-LD) and the Australian Defense Force Academy Windows Dataset (ADFA-WD), which are newly developed systems that identify datasets containing vulnerabilities and address attacks on diverse products. The simulation demonstrates this by employing a single hidden layer and a limited number of nodes.

In [21], the applications of ML algorithms in IoT security are examined by evaluating numerous studies that have used various techniques to solve a number of issues affecting the security of the IoT environment. This study emphasizes the scope of the IoT and the opportunities for its use. The authors then concentrate on the specific cases in which ML might bolster IoT security, such as the use of malware and intrusion detection and the recognition of unidentified IoT devices.

A statistical solution is used in [22]. In this study, CUSUM and EWMA are tested using the same dataset, and EWMA outperforms CUSUM, as expected, because EWMA gives additional weight to more recent data. The most significant aspect for us is the fact that both algorithms produce favorable results and successfully detect and mitigate attacks, which is one of the reasons we chose them for our paper.

In this work, we have used a combination of three algorithms (CUSUM, EWMA, and KNN). We have achieved a high accuracy, or 99.00%, with a low false positive rate (FPR).

All the previous works mentioned above were classified into three types: machine learning, knowledge-based systems, and statistics. The approach presented in this research is one type of integration for all of them. This approach has overcome the limitations of the previous studies by its ability to implement a mitigation system that can detect the attack with high accuracy and a low FPR, in addition, to its ability to distinguish between IoT and non-IoT devices.

3. Methodology

3.1. Dataset Description

The Bot-IoT dataset was established at the University of New South Wales (UNSW) in Australia [23]. This dataset includes both legitimate and simulated IoT traffic, as well as various types of attacks. The Bot-IoT dataset is classified into three main sub-categories, as shown in Figure 2. The dataset accurately depicts the traffic flow features of the situation in which the framework would be deployed. This dataset will be used for the experiments. For our experiment, we use the Bot-IoT dataset because:

- Compared with other datasets, it is the most recent and advanced dataset with simulated IoT services.
- It contains a variety of common IoT attack samples with significant data features.
- It contains modern networking features.

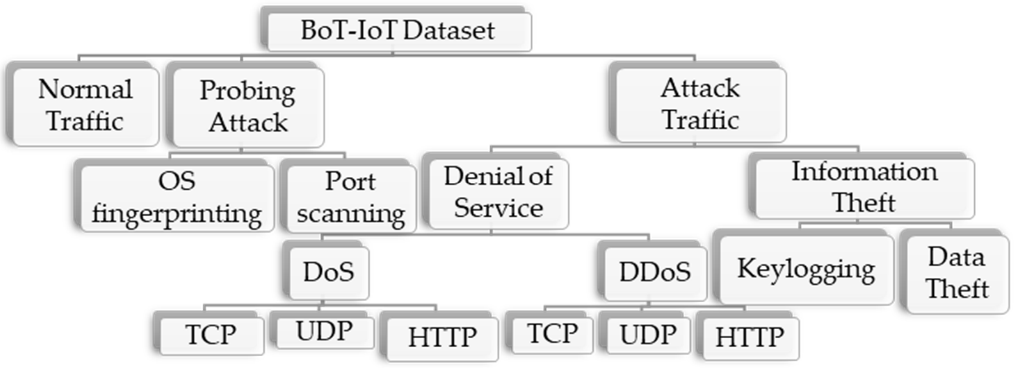


Figure 2. Bot-IoT dataset classification.

3.2. Methods of Feature Selection

Feature selection is a machine learning (ML) technique for reducing the number of input characteristics to increase the design model performance, while reducing memory and CPU processing costs. This method can also be used to simplify the design models so that they are simpler to understand and evaluate the level of the chosen characteristic. Therefore, we use two techniques for selecting features such as entropy, correlation coefficient, synthetic minority over-sampling technique (SMOTE), and SHAPley Additive exPlanations (SHAP), described as follows:

- Entropy: it describes the vulnerability of or confusion between elements in the Bot-IoT dataset [24].
- Correlation coefficient: It is used for estimating the straight connection between the list of capabilities of the Bot-IoT dataset. Its result range is [-1, 1], its extent demonstrates the power of connection between two element vectors, and its sign shows the correlation of relationship, that is, positive or negative. [25]. Both entropy and correlation coefficient are used to select the 10 best features.
- SMOTE: the over-sampling principally includes expanding the size of an imbalanced dataset by copying the minority of occurrences of certain classes that are equally distributed among the classes.
- SHAP: SHAP values are employed [26] to recognize the main elements that influence the result of our model classifier. SHAP, or SHAPley Additive exPlanations, is a game hypothetical methodology that can be used for making an ML model more reasonable by imagining its result. It tends to be used for making sense of the expectation of any model by figuring out the commitment of each component to the expectation. We use

this technique to acquire data on the performance of our model classifier and explain it compared to the dataset used.

To analyze the correlation between features, we compute a correlation matrix. In Figure 3, a heatmap is plotted to view the determined connection values between various elements. This map displays the degree to which both elements are directly connected. The correlation coefficient can have values ranging from −1 to 1. A value of nothing or near zero shows that no relationship exists between two elements. A value of 1 or near –1 demonstrates that the two elements are profoundly correlated.

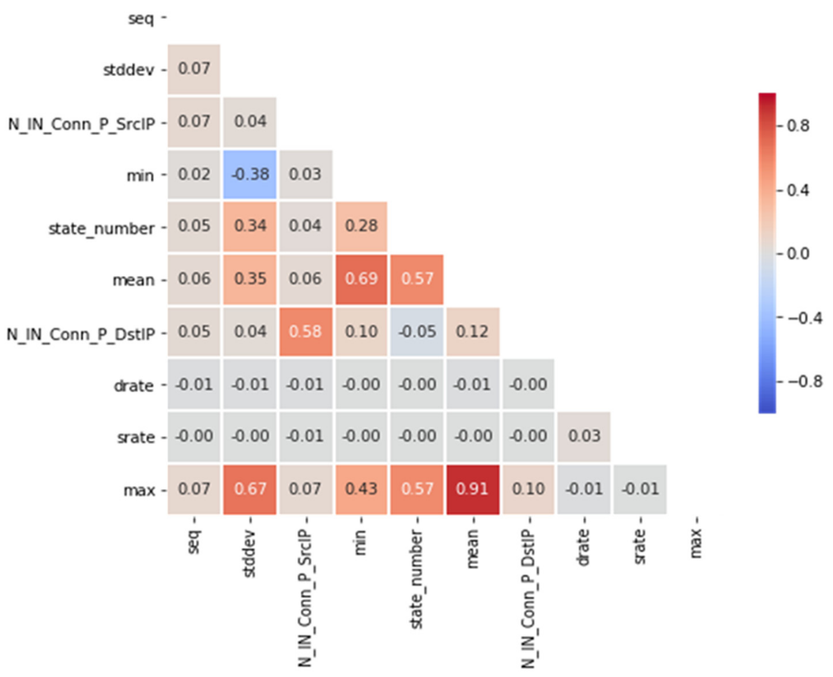


Figure 3. Heatmap in view of the relationship between elements.

The plot in Figure 4 shows that the most important feature is N_IN_Conn_P_SrcIP (the number of incoming associations per source IP), and there is a big difference between this feature and the other important ones based on the F1 score. The feature mean has the lowest importance; therefore, we eliminate it before building our models. This graph helps us to choose parameters that will be used to train the models.

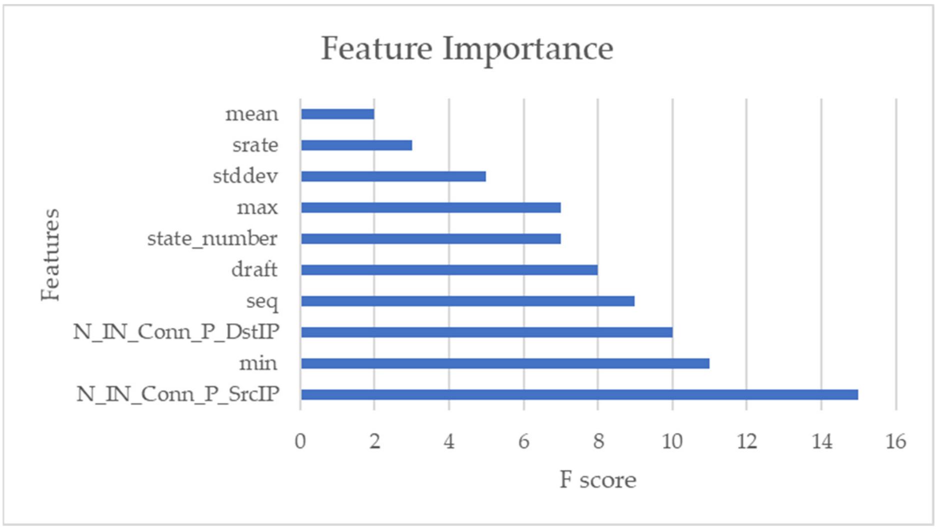


Figure 4. The feature importance score.

3.3. Dataset Pre-Processing

Before evaluating the performance of any categorization algorithm, dataset pre-processing is essential, particularly for ML-based mitigation schemes, to improve categorization accuracy and avoid misleading results. Figure 5 depicts the pre-processing steps we used on the datasets, including data cleaning, data transformation, feature selection, and class distribution balancing. We used 5% of the dataset. It includes the top ten features for training and testing datasets. This dataset has 19 columns, with target columns of classification and sub-category as the objective sections, and over 2.9 million rows [23].

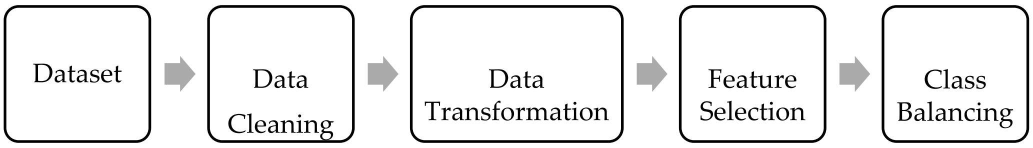


Figure 5. Data pre-processing steps.

3.4. Categorization Using ML Models

In this research we have trained nine ML models: logistic regression (LR), support victor machine (SVM), XGBoost, random forest (RF), naive bayes (NB), decision tree (DT), K-nearest neighbors (KNN), exponentially weighted moving average (EWMA), and the cumulative sum algorithm (CUSUM). The result shows that KNN provided the best average score for all the parameters. In our proposed solution, we have combined three algorithms: KNN, EWMA, and CUSUM.

3.5. Mathematical Model

A mathematical model can be used to estimate the system's quantitative behavior. The quantitative conclusions from statistical pattern can be simply contrasted with experimental data to determine the model's strengths and limitations [27]. As a result, in this part, a statistical pattern for detecting DDoS attacks is proposed. The most important qualities for determining attacks are accuracy and the false positive rate [28]. In our proposed solution, we have achieved a high accuracy and a low FPR by combining three algorithms (KNN, EWMA, and CUSUM), rather than implementing each algorithm individually. In the next section, the paper explains each of these algorithms in detail.

3.5.1. Cumulative Sum (CUSUM) Algorithm

This algorithm has the ability to detect changes. CUSUM is modernized on a regular basis, which is useful for this implementation solution since network traffic typically contains a large number of packages that are continually modifying throughout their period cycle. It is a quality control algorithm that is enhanced for monitoring any variation from a predefined value and for detecting tiny mean changes. CUSUM calculates the total number of variances among both the actual and expected principles, which is the CUSUM value. CUSUM is extremely adaptable and is currently implemented in several applications. It can even be bestowed with an independent learning ability, allowing it to mitigate and detect variations in network traffic [29]. This algorithm is the average of the variation in a signal μ, which is a real-time rate updated on a regular basis. Furthermore, if S_i is the i-th CUSUM, x_n is the n-th monitoring, and μ is the real-time average of the procedure, the CUSUM S_i can be determined as shown in Equation (1):

$$S_i = \sum_{i=1}^{n} (x_1 - \mu) = (x_n - \mu) + S_i - 1 \tag{1}$$

3.5.2. Exponentially Weighted Moving Average (EWMA)

EWMA uses exponentially decreasing weighting elements. The previous data are much less valuable than the current data, but they are still considered [30]. This feature is important since an attack can occur, and then be immediately stopped, making the current data additionally crucial. A continuous smoothing component b, a value between 0 and 1, is used to describe the degree of weighing reduction. The value of b can alternatively be stated as a percentage. EWMA is calculated using Equation (2):

$$\overline{\mu_n} = \beta \overline{\mu_n} - 1 + (1 - \beta) \tag{2}$$

where μ_n is the average of the procedure assessed in reality in the n-th monitoring and the β is the EWMA component.

3.5.3. K-Nearest Neighbors (KNN)

The KNN approach is a simple method used for both regression and classification. It is a non-parametric method, which is advantageous since in real-world settings, whenever it comes to attacking data, there will often be no regulations. Dependent upon K number of training examples, this method classifies or predicts [31], for example, by relating other entered data to the data of a similar class as the nearest number of K examples for a given value of K. There are numerous techniques for calculating distance to the nearest instances. However, the greatest common factor is called the Euclidean distance, which is determined as shown in Equation (3):

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3}$$

where x_i is the i-th monitoring and y_i is the i-th observation.

Table 1 compares the three algorithms (KNN, EWMA, and CUSUM).

Table 1. Comparison of EWMA, CUSUM, and KNN.

EWMA	CUSUM	KNN
Works well when the rate of attacks is high.	Is effective in detection using a statistical approach.	Is highly effective and trainable.
Is effective in detection using a statistical approach.	Detects attacks successfully.	Is simpler and more productive.
Is able to highlight relatively new data by giving the data more weight.	Is used for excellent control.	Can be used for both regression and classification.
Detects attacks successfully.	Is designed to measure any deviation from a given amount and recognizes tiny mean changes.	

Algorithm 1 shows that our proposed solution has the advantage of having its own backup system, where, if one of the three algorithms fails to identify malicious traffic, the two other algorithms will succeed. For the same two values, the results of three algorithms, EWMA, CUSUM, and KNN, remain associated. The potential outcomes are 0 (an attack is taking place) and 1 (no attack is taking place). Initially, the results of EWMA and CUSUM are compared and if they are equal, the result of CUSUM is considered to be the final result, but if they differ, an additional differentiation among CUSUM and KNN is performed, yielding a final output. If the result of KNN matches that of CUSUM, we take the CUSUM principle as the output. Otherwise, we take EWMA or KNN as the output because EWMA and KNN hold equal principles (that is, the conflicting of the CUSUM principle), which is what we need in this paper. Then, we can decide whether or not to raise an alarm. Figure 6 presents the complete attack and anomaly detection procedure. After the dataset is loaded and the type of attack identified, the dataset passes through different data pre-processing steps. Then, the dataset splits into training and testing data. The different ML algorithms are applied to measure and evaluate the model against multiple measurements. A final model is generated using the novel multialgorithm implemented in this research, and the measurements are compared.

Algorithm 1: Proposed Algorithm Advantages of Backup

```
1: Input: CUSUM, EWMA, KNN
2: Output: Output
3: Run Log–CUSUM (Output-CUSUM)
4: Run Log–EWMA (Output-EWMA)
5: Run Log–KNN (Output-KNN)
6: If Output–CUSUM = Output–EWMA then
7:     Output = Output-CUSUM;
8: else
9:     If Output–CUSUM = Output–KNN then
10:        Output = Output-CUSUM;
11:    else
12:            Output = Output-EWMA;
```

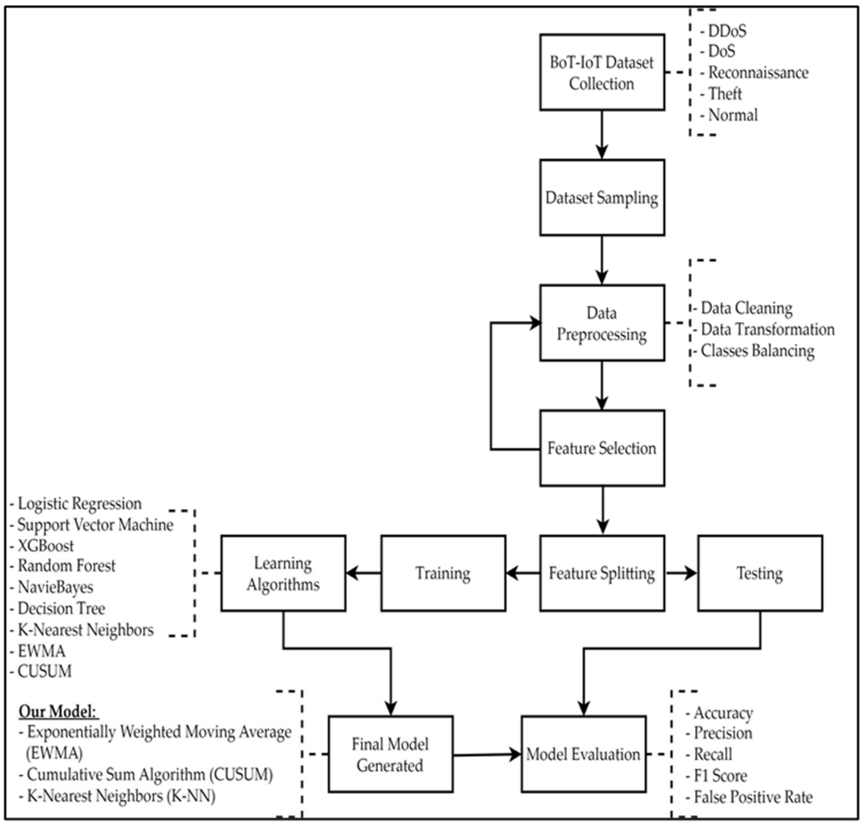


Figure 6. Complete attack and anomaly detection procedure.

3.6. The Proposed Framework

Figure 7 presents the framework solution proposed in this paper. By characterizing and profiling network traffic traces, we sort the incoming traffic to isolate the IoT device from the non-IoT device. This designed system is able to distinguish between the traffic generated by IoT gadgets and the traffic generated by non-IoT gadgets. The IoT traffic is forwarded to detection engine 1, and the non-IoT traffic is forwarded to detection engine 2. Then, we categorize the traffic into normal, suspicious, and confirmed attack traffic. Our proposed framework monitors the process over time and signals an alarm if it detects abnormal behavior. The normal traffic is forwarded to the Internet, the suspicious traffic is investigated by a second checkpoint to determine whether or not it is an attack, and the confirmed attack traffic is dropped, and the source of this traffic is blacklisted. This research focuses on TCP SYN flood.

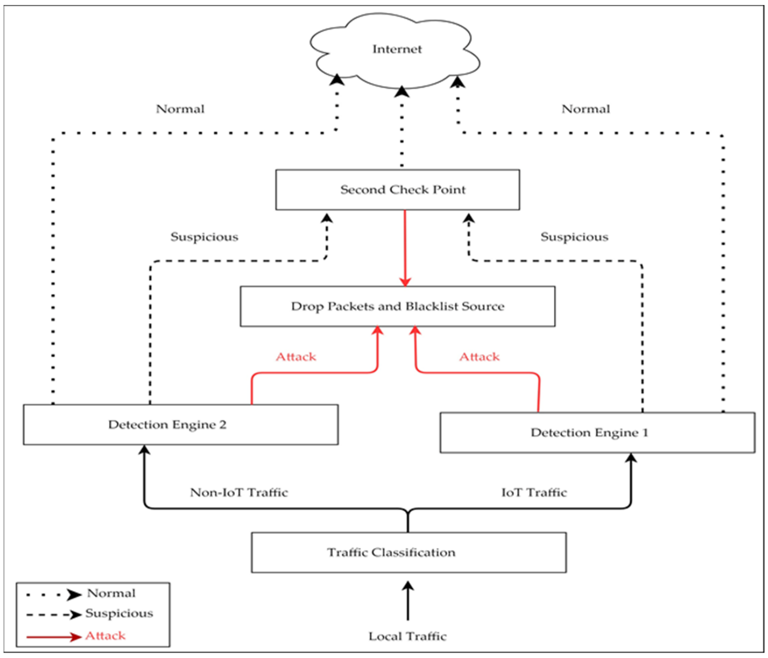


Figure 7. The proposed framework.

The model in Figure 8 identifies the traffic and distinguishes between IoT devices and non-IoT devices. We upload the dataset [23] in CSV file format. The network traffic data are 8 GB in size, sufficient for reasonable results. The network has 20 connected devices, including both IoT and non-IoT devices. We use the Wireshark tool to show the network traffic traces and capture the network traffic packets, as shown in Figure 9. The flow features are organized as tuples that include time, packet size, protocol, source MAC, destination MAC, destination port source IP, source port, destination IP, and information approved through packets, which are all factors to be considered.

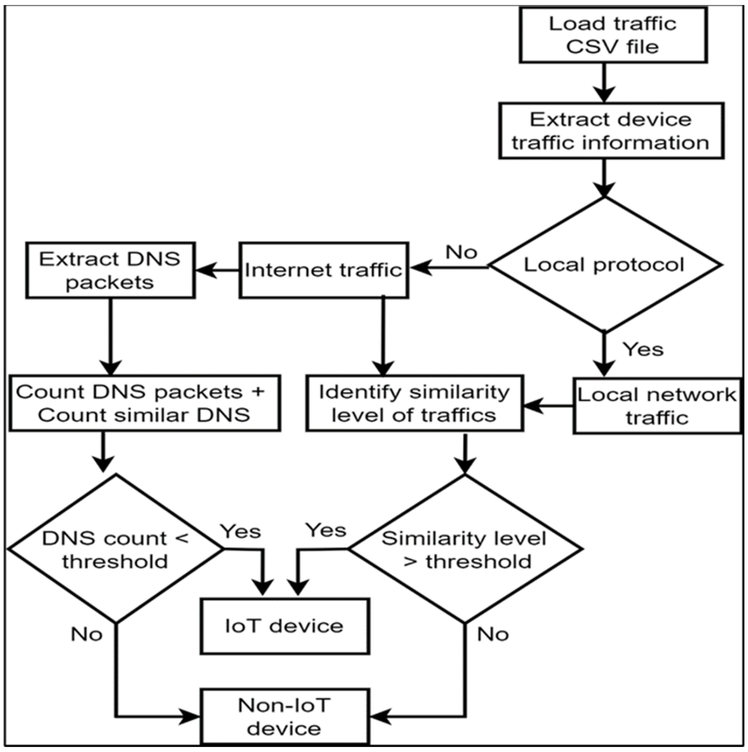
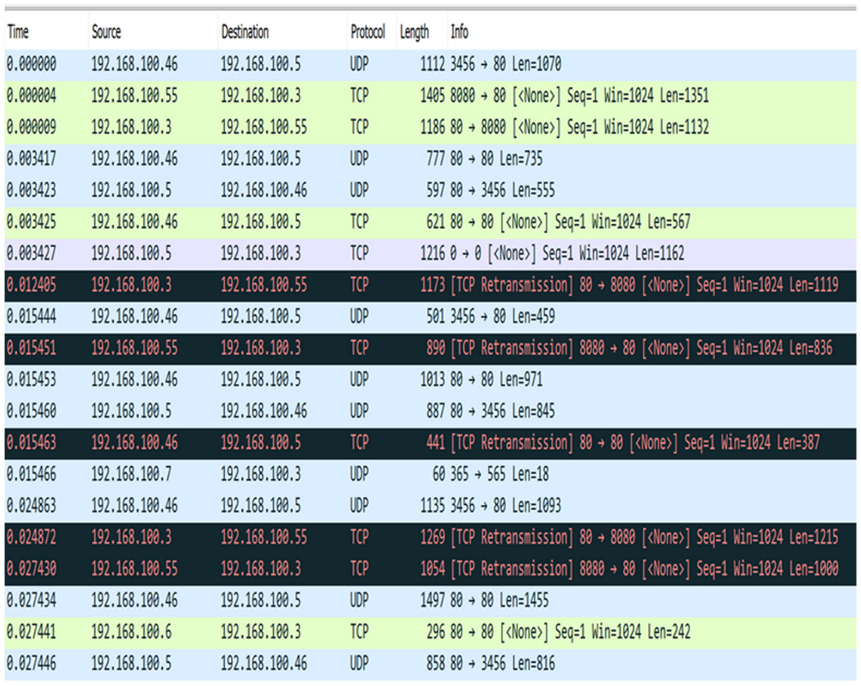


Figure 8. Model for distinguishing between IoT devices and non-IoT devices.



Time	Source	Destination	Protocol	Length	Info
0.000000	192.168.100.46	192.168.100.5	UDP	1112	3456 → 80 Len=1070
0.000004	192.168.100.55	192.168.100.3	TCP	1405	8080 → 80 [<None>] Seq=1 Win=1024 Len=1351
0.000009	192.168.100.3	192.168.100.55	TCP	1186	80 → 8080 [<None>] Seq=1 Win=1024 Len=1132
0.003417	192.168.100.46	192.168.100.5	UDP	777	80 → 80 Len=735
0.003423	192.168.100.5	192.168.100.46	UDP	597	80 → 3456 Len=555
0.003425	192.168.100.46	192.168.100.5	TCP	621	80 → 80 [<None>] Seq=1 Win=1024 Len=567
0.003427	192.168.100.5	192.168.100.3	TCP	1216	0 → 0 [<None>] Seq=1 Win=1024 Len=1162
0.012405	192.168.100.3	192.168.100.55	TCP	1173	[TCP Retransmission] 80 → 8080 [<None>] Seq=1 Win=1024 Len=1119
0.015444	192.168.100.46	192.168.100.5	UDP	501	3456 → 80 Len=459
0.015451	192.168.100.55	192.168.100.3	TCP	890	[TCP Retransmission] 8080 → 80 [<None>] Seq=1 Win=1024 Len=836
0.015453	192.168.100.46	192.168.100.5	UDP	1013	80 → 80 Len=971
0.015460	192.168.100.5	192.168.100.46	UDP	887	80 → 3456 Len=845
0.015463	192.168.100.46	192.168.100.5	TCP	441	[TCP Retransmission] 80 → 80 [<None>] Seq=1 Win=1024 Len=387
0.015466	192.168.100.7	192.168.100.3	UDP	60	365 → 565 Len=18
0.024863	192.168.100.46	192.168.100.5	UDP	1135	3456 → 80 Len=1093
0.024872	192.168.100.3	192.168.100.55	TCP	1269	[TCP Retransmission] 80 → 8080 [<None>] Seq=1 Win=1024 Len=1215
0.027430	192.168.100.55	192.168.100.3	TCP	1054	[TCP Retransmission] 8080 → 80 [<None>] Seq=1 Win=1024 Len=1000
0.027434	192.168.100.46	192.168.100.5	UDP	1497	80 → 80 Len=1455
0.027441	192.168.100.6	192.168.100.3	TCP	296	80 → 80 [<None>] Seq=1 Win=1024 Len=242
0.027446	192.168.100.5	192.168.100.46	UDP	858	80 → 3456 Len=816

Figure 9. Network traffic traces using the Wireshark tool.

After the entire device traffic has been divided, local and web traffic packets must be differentiated. On the basis of the packet protocol, Python is used to separate the Domain Name Server (DNS) packets from individual gadgets in the web traffic. Then, the DNS packets between dissimilar domains and comparable (repeated) domain packets are checked and counted. When the IoT DNS packets have been examined in detail, we discovered that a large number of packets have the same destination domain and a similar packet size and query. We also discovered that a few devices transmit comparable packets in a normal period on a similar domain, indicating that IoT gadgets transmit restricted DNS packets to a limited domain.

In the research, each device is individually iteratively inspected, and the dataset shows that IoT devices transmit DNS requests to an extreme of 10 diverse domains. Therefore, we establish a DNS count threshold of 10. In contrast, the script calculates the number of frequent packets of device traffic through checking packet destination port, IP, and protocol size. When these characteristics have similar principles, it indicates that this gadget sends out frequent or comparable packets. When the gadget comprises at least 25% of the frequent packets, it is likely that this is an IoT gadget. This principle is obtained through an iterative method of inspecting the traffic of various devices. We conclude that a minimum of 25% of the IoT device packets must be repeated. We discover DNS query designs that are obviously distinct from non-IoT devices. The traffic patterns of some devices are consistent, indicating that as soon as a user uses a gadget, it produces the equivalent action sequence, which provides an expectable outcome. The research findings help the network administrators to better distinguish between IoT and non-IoT devices, allowing them to create higher quality network policies in terms of security, routing, and resource distribution.

This part introduces the proposed framework, discusses a use case for the fog computing framework in the context of anomaly mitigation, and provides a brief overview of the components of the framework. A framework based on the fog computing paradigm is presented to detect and mitigate anomalies, for instance, botnet attacks in the IoT network, as seen in Figure 10. The fog computing paradigm is used in IoT networks to compensate for a shortage of resources [32]. It relieves resource-constrained IoT devices of the load of mathematical overhead and additional relevant functioning needs in the mitigation of an anomaly. The framework uses an anomaly-based IDS module that uses blacklisted IP addresses stored in a database and an algorithm classifier that we have implemented in our solution. This allows the detecting module to take advantage of their strengths. The anomaly-based approach detects and mitigates attacks with reasonable accuracy. This framework guarantees a secure IoT network.

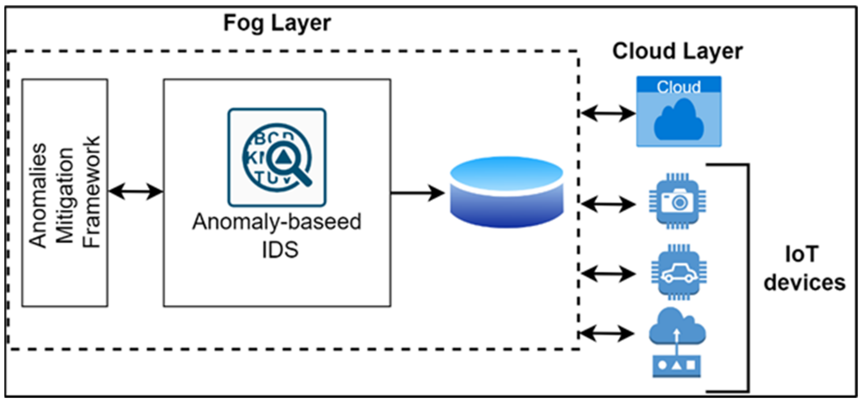


Figure 10. Proposed fog framework.

In the proposed framework, in Figure 10, x represents the network traffic flow, d represents the known attacks (from the blacklisted IP addresses), N represents the normal traffic stream, and A represents the irregular traffic stream. First, the network flow x passes through the anomaly-based IDS module. The IP address of the traffic flow x is checked in contrast to d in the module's database. If $x \in d$ is true, x is blocked/dropped and an alarm is raised for the manager admin. If not, the stream is redirected to the second check point. The anomaly-based detection module classifies the network traffic flow x as N or A. The module blocks/drops x if it is categorized as A. After that, an alarm is raised and forwarded to the manager admin. Lastly, the IP address of A is changed in the database of the anomaly-based detection module. The x, on the other hand, is allowed to pass.

The proposed fog framework in Figure 11 can be used to create a fog node on a dedicated system to efficiently implement security solutions such as the anomaly mitigation system IDS to protect the IoT node or system, ensuring that malicious users cannot use the IoT nodes as a botnet part to carry out DDoS attacks.

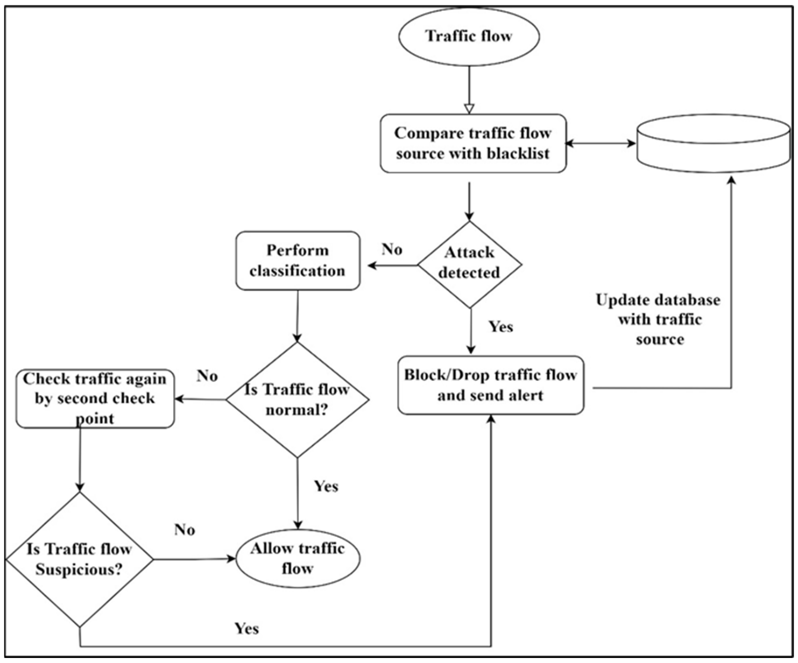

Traffic flow
Compare traffic flow source with blacklist
Attack detected
No
Yes
Perform classification
Block/Drop traffic flow and send alert
Update database with traffic source
Is Traffic flow normal?
No
Yes
Check traffic again by second check point
Is Traffic flow Suspicious?
No
Allow traffic flow
Yes

Figure 11. The operational flow chart of the proposed framework.

The traffic classification step is performed using Algorithm 2, which works well in distinguishing between traffic types.

Algorithm 2: Proposed Anomaly Mitigation Algorithm

```
1: Input: Traffic flow
2: Output: Attack detected
3: Step1: Set
4: T ← Traffic flow
5: Tip ← Traffic Source IP
6: T f ← Traffic Features
7: BL ← Blacklist
8: Step2: Blacklist look-up
9: Get Tip
10: if Tip ∈ BL then
11:     Block/drop Traffic flow
12:     Send alert
13: else
14:     Go to Step3
15:     Step3: Classification
16:     Start classification model (CUSUM)
17:     Input: T f
18:     Output: Normal or Suspicious or Attack
19:     if T = Attack then
20:         Block/drop Traffic flow
21:         Send alert
22:         Update database
23:     else if T = Normal then
24:         Allow Traffic flow
25:     else T = Suspicious then
26:         Allow Traffic to pass to Second Check Point
27:         if T = Suspicious then
28:             Block/drop Traffic flow
29:             Update database
30:         else
31:             Allow Traffic flow
32:             end
```

In summary, the methodology used in this study consists of the following steps: First, the dataset used in this study is described. Then, the features selected for implementation are identified. After that, the dataset pre-processing steps are followed. Then, different ML model algorithms are implemented. After that, a mathematical model is presented to describe the three algorithms used in the proposed solution. Then, the framework used for the proposed solution is provided. After that, a model to distinguish between IoT and non-IoT devices is presented. Finally, a framework based on the fog computing paradigm is presented to detect and mitigate anomalies.

4. Evaluation Approach

4.1. Experimental Details

This part describes the research experimental environment design, the evaluation metrics, and the performance examination for the proposed method. For this evaluation experiment, we use the Jupyter Notebook with the Python-based Scikit-learn ML libraries on a Windows 10 operating system with 16 GB of RAM and an Intel® Core ™ i7-8650U CPU at 2.11 GHz. Our model requires Matplotlib, Scipy, Pandas, Detecta, and Scikit-learn. We will assess the proposed categorization algorithms from two aspects:

- Pre-processing stages.
- Traffic categorization.

The proposed framework is executed using the Python programming language. The obtained dataset is used as the source of data for the proposed framework. To evaluate the anomaly-based module, the obtained dataset is reproduced as two duplicates and will involve many data pre-processing procedures, including data transformation (encoding) and feature selection (correlation coefficient and entropy techniques, the synthetic minority over-sampling technique (SMOTE), and SHAPley Additive exPlanations (SHAP)) [14]. In each iteration of testing sets, a random seed [28] will be used for the Stratified K Fold purpose to create distinguishable datasets for training and testing, while preserving the equivalent proportion of the aim labels in the training and testing sets to ensure accurate results for all the classes when we are evaluating our model. K-fold cross validation will be employed with k = 10. The 10-fold cross-validation randomly splits the dataset into 10 parts. In every examination, one part of the split dataset will be used as a test set, while the remaining nine parts will be used as a training set.

4.2. The Requirement of the Experiment

- Python: It is an open-source, high-level, general-purpose programming language. It has been the preferred programming language for the majority of data scientists due to its code readability philosophy, efficient code, easy communication features, and ease of learning. Python has vibrant scientific libraries, as well as a variety of fantastic environments, such as Spyder and Jupyter Notebook. Python Matplotlib is a powerful 2D graphic library that aids machine learning scientists in graph plotting. Because of these advantages, we chose Python languages for our machine learning experiment.
- Scikit-Learn: It is a public and open-source ML library for Python. It is a quick and easy tool for data mining and analysis. Scikit-learn contains implementations of various supervised and unsupervised learning algorithms. This package includes features for model selection, dimensionality reduction, and data pre-processing, in addition to classification, regression, and clustering algorithms.
- SciPy: SciPy stands for Scientific Python and is a technical calculation library that uses NumPy underneath. It is an open source library, so we can use it freely. Optimization, linear algebra, integration, interpolation, and special functions are all modules in SciPy.
- Matplotlib: It is a plotting library for Python, as well as its numeral math extension, NumPy. It offers an object-oriented API for incorporating plots into apps that use general-purpose GUI toolkits.
- Pandas: It is an analysis software library written for Python and data manipulation. It offers operation and data structures for manipulating numeral tables and time series.
- Detecta: It is Python module to detect proceedings in data. In our implementation, detecta_cusum is used to detect unexpected changes in data using CUSUM.

4.3. The Evaluation Metrics

There are five performance metrics widely used in the literature: accuracy, recall, precision, F1 score, and the FPR [33,34]. This research will use these metrics to evaluate the proposed anomaly mitigation scheme. This section presents the methods used to examine the research model.

Accuracy: It is the proportion of correct detections to false alerts produced by an IDS model, indicating the total achievement rate of any IDS. Equation (4) is used to calculate accuracy.

$$Accuracy = (TN + TP)/(TP + FP + TN + FN) \tag{4}$$

Recall: It is also known as the true positive rate (TPR) or DR. It is the percentage of malicious vectors that have been correctly classified out of the overall number of malicious vectors. Equation (5) is used to calculate recall.

$$Recall = TP/(FN + TP) \tag{5}$$

Precision: It is also known as the false negative rate (FNR). It describes the proportion of incorrectly classified attacks in comparison to the overall number of attack examples. Precision can be calculated using Equation (6).

$$Precision = FN/(FN + TP) \tag{6}$$

*F*1 *Score*: It is the calculated weighted average of both the DR and the FNR. The F1 score can be calculated using Equation (7).

$$F1\ Score = 2{*}(Recall{*}Precision)/(Recall + Precision) \tag{7}$$

False Positive Rate (FPR): It is the proportion of detections incorrectly classified as attacks to the overall number of detections. Equation (8) is used to calculate the *FPR*.

$$FPR = FP/(FP + TN) \tag{8}$$

Here, we describe the equation of the evaluation metrics: TP is the number of attack samples that are categorized properly, TN is the number of normal samples that are categorized properly, FP is the number of attack samples that are categorized inaccurately, and FN is the number of normal examples that are categorized inaccurately. TPR or recall is the number of attack samples that are correctly categorized among the overall attack examples, TNR is the number of normal samples that are correctly categorized among the overall normal examples, FPR is the number of attack examples that are inaccurately categorized among the overall normal examples, and FNR is the number of normal examples that are inaccurately categorized among the overall attack examples. Accuracy is the overall number of categorized samples divided by the overall number of samples saved in the dataset. Precision is the proportion of correctly categorized samples among the overall number of correct samples.

4.4. The Experimental Results

The goals of the proposed solution were to detect an attack in an early stage and inform the user and decrease the false positive rate (FPR). To achieve these goals, we combined three algorithms: exponentially weighted moving average (EWMA), K-nearest neighbors (KNN), and the cumulative sum algorithm (CUSUM).

Figure 12 presents our calculation of the accuracy, recall, precision, and F1 score of our model compared with those of other algorithms.

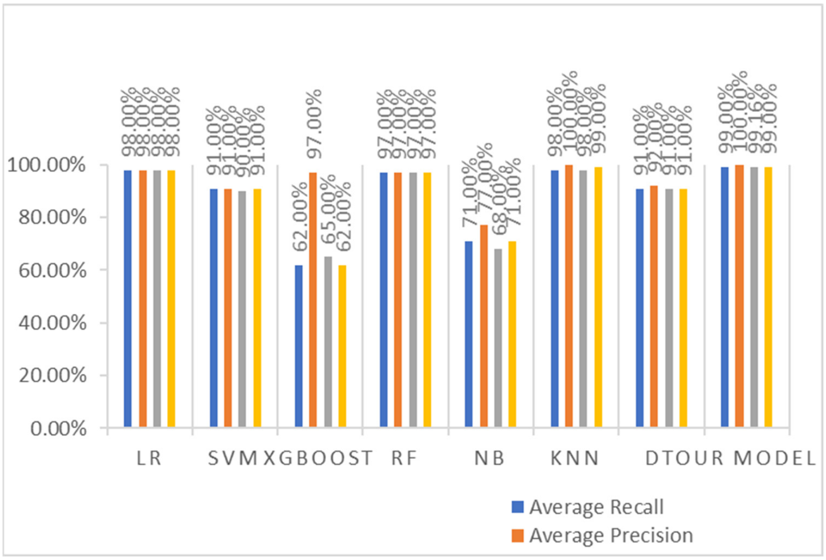


Figure 12. Comparison between different performance averages.

In Appendix A, we present our calculation of the precision, recall, F1 score and accuracy of our model compared with those of other algorithms. Figure A1 presents a comparison of the precision of our model with different algorithms. Figure A2 presents a comparison of the recall of our model with different algorithms. Figure A3 presents a comparison of the F1 score of our model with different algorithms. Figure A4 presents a comparison of the accuracy of our model with different algorithms.

Figure 13 presents our calculation of the accuracy, recall, precision, and F1 score of the three selected algorithms (EWMA, CUSUM, and KNN).

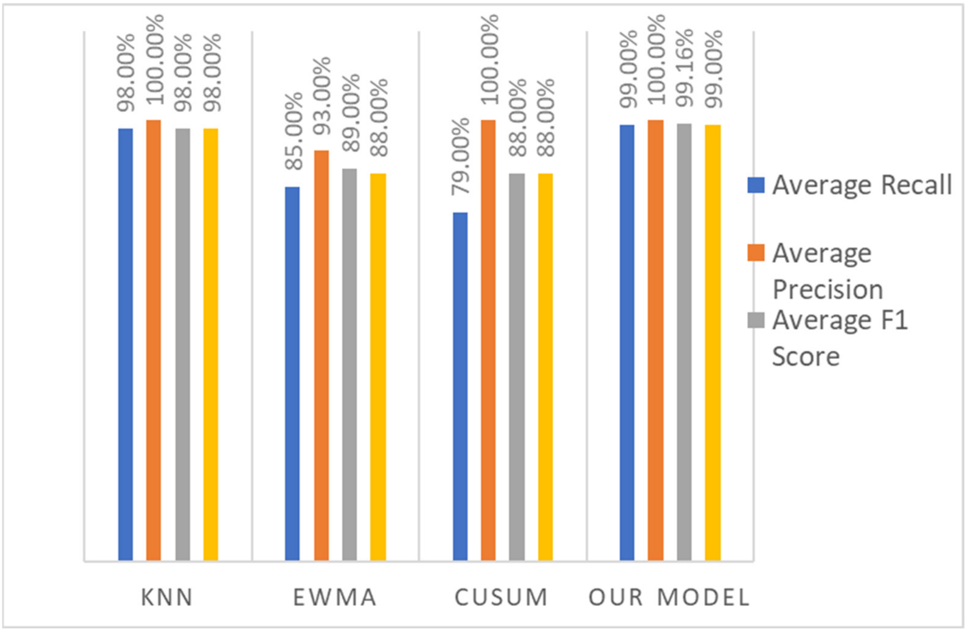


Figure 13. Average performance of EWMA, CUSUM, and KNN.

In Appendix A, Figure A5 presents a comparison of the recall of EWMA, CUSUM, and KNN with our model. Figure A6 presents a comparison of the precision of EWMA, CUSUM, and KNN with our model. Figure A7 presents a comparison of the F1 score of EWMA, CUSUM, and KNN with our model. Figure A8 presents a comparison of the accuracy of EWMA, CUSUM, and KNN with our model.

Figure 14 presents our calculation of the false positive rate (FPR) of our model compared with the FPRs of other algorithms.

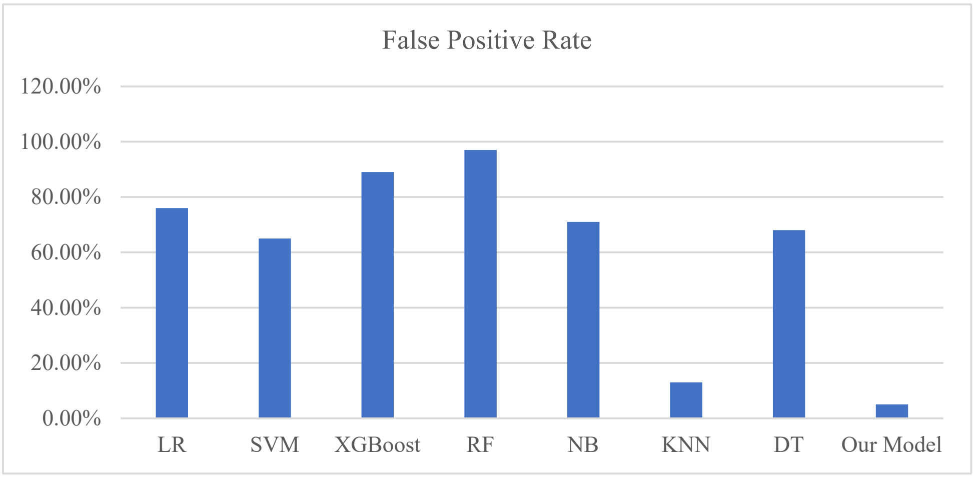


Figure 14. Comparison of the false positive rate with different algorithms.

In Appendix A, Figure A9 presents a comparison of the FPR of EWMA, CUSUM, and KNN with our model.

Figure 15 presents the network traffic expressed as an amount of packets, and the period is expressed in seconds.

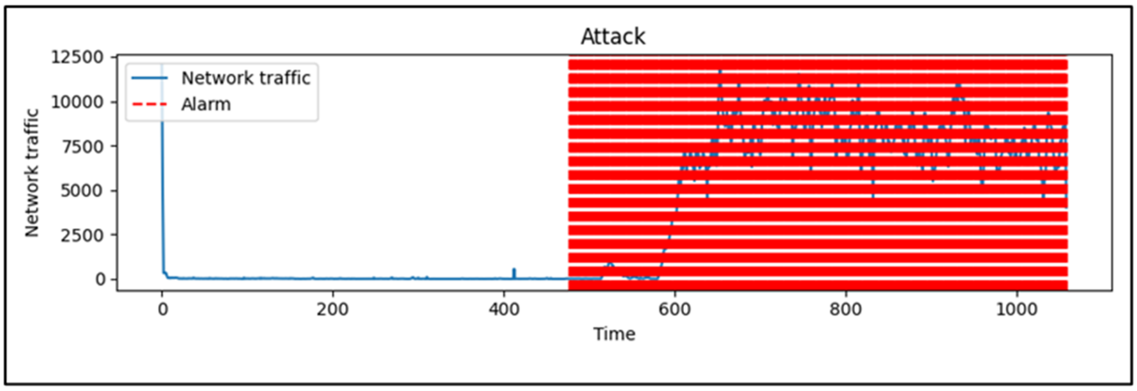


Figure 15. Network flooding attack traffic.

Figure 16 presents how the KNN algorithm detects malicious traffic.

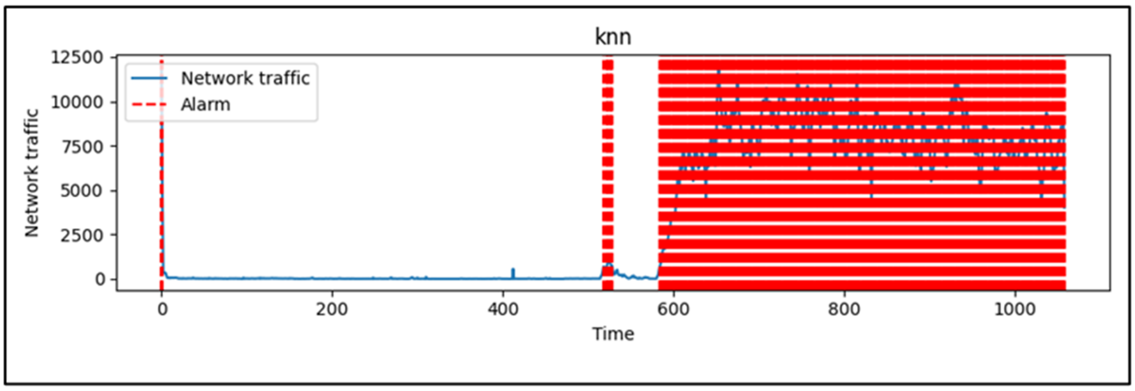


Figure 16. Flooding attack detection by KNN.

Figure 17 shows how the exponentially weighted moving average algorithm detects malicious traffic.

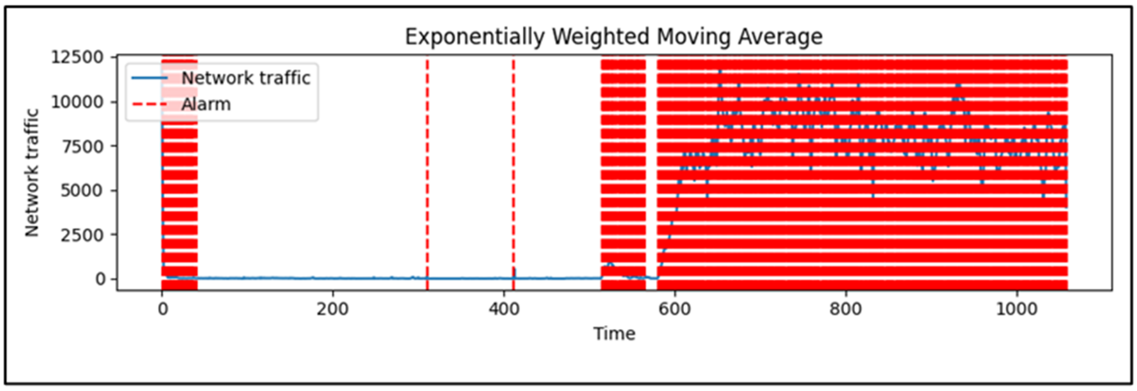


Figure 17. Flooding attack detection by EWMA.

Figure 18 presents how the cumulative sum algorithm detects malicious traffic.

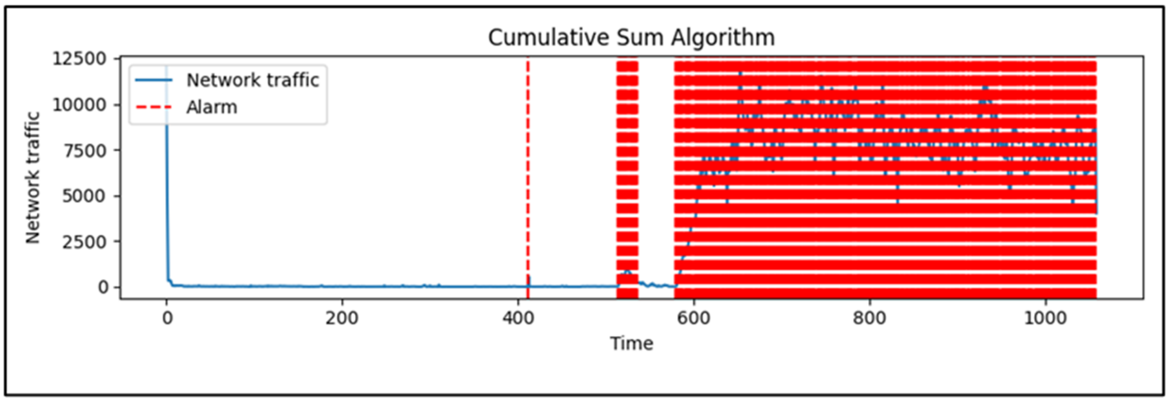


Figure 18. Flooding attack detection by CUSUM.

Figure 19 shows how the three algorithms (EWMA, CUSUM, and KNN) that we have used in our model detect malicious traffic.

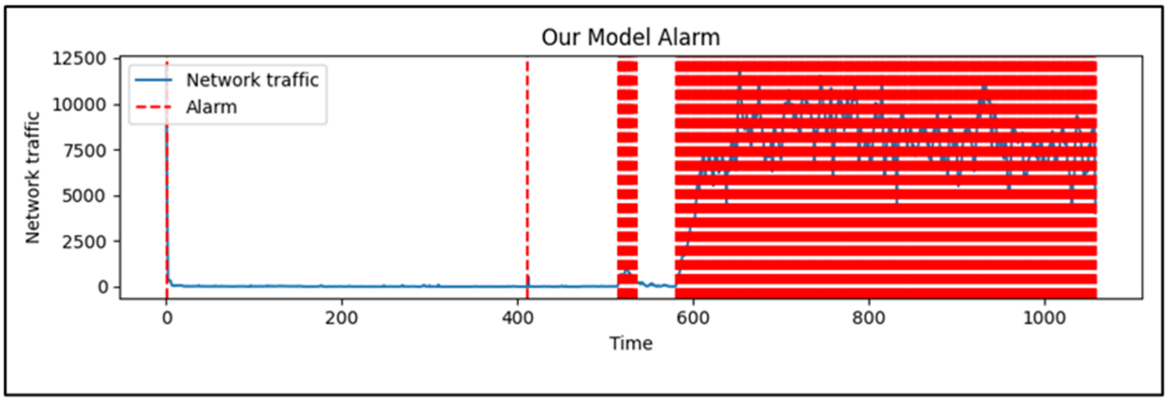


Figure 19. Flooding attack detection by our model (EWMA, CUSUM, and KNN).

EWMA, KNN, and CUSUM have been tested to determine all the benefits of the proposed implementation. First, the data are marked to ensure that the algorithms can be contrasted to several benchmark principles. For example, in Figure 15, the traffic of the network is represented by the number of packages, and time is represented in seconds.

When one is identifying malicious traffic, different metrics are used for measuring accuracy, precision, F1, and DR for EWMA, KNN, and CUSUM, and one should examine and compare them. In our model, we have combined three algorithms to obtain the maximum advantages of these three algorithms at once. Figures 16–18 depicts how the KNN, EWMA, and CUSUM algorithms individually identify malicious attack traffic, respectively. However, Figure 19 shows how the three algorithms detect malicious traffic when they are combined, as in our model.

Overall, our model was the one that achieved the most robust results when we were measuring all the metrics used in this thesis, such as precision, accuracy, F1, and recall, by combining the three algorithms, rather than using any of the three algorithms individually.

4.5. Comparison with Other Approaches

We compared the performance of our model with other studies as shown in (Table 2) and found that our model is more effective in detecting attacks in an early stage with a high rate of accuracy and a low FPR.

Table 2. Comparison of the implemented work with related studies.

Study	Solution	CUSUM	EWMA	KNN	Anomaly Detection	Adaptive
[7]	Entropy	x			x	
[8]	IDS CUSUM	x			x	x
[9]	CUSUM Flood	x			x	
[22]	Statistical	x	x		x	x
This work	Our model	x	x	x	x	x

5. Conclusions

In conclusion, the integration of fog computing with the Internet of Things has created an effective platform for implementing anomaly mitigation strategies to address security issues such as botnet threats. We used the Bot-IoT dataset to examine the proposed modules. From the result, we conclude that our model, with a combination of algorithms, has achieved better results in terms of accuracy, precision, recall, and F1 score than any other algorithms individually did. We have achieved a higher accuracy score with a low false positive rate (FPR). Our solution can be used in real-time systems and be integrated with other software solutions. The limitation of the solution implemented is the time of execution because of the use of three algorithms. However, we have achieved the goal of this research by attaining a high rate of accuracy (99.00%) with a low FPR. We have also achieved a good result in terms of distinguishing between IoT and non-IoT devices, which will help networking teams to make the same distinction.

Future studies can:

- Examine the performance of different types of ML algorithms (unsupervised and deep learning (DL)).
- Make a dataset with real-world IoT devices.
- Compare the impact of performance metrics with and without balancing the dataset.
- Focus on other DDoS attack types to protect IoT services.

Author Contributions: Conceptualization, R.J.A.; funding acquisition, R.J.A.; methodology, R.J.A. and A.A.; resources, R.J.A.; supervision, A.A.; visualization, R.J.A.; writing—original draft, R.J.A.; writing—review and editing, R.J.A. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by King Abdulaziz University under grant No.(4-15 1432/HICI).

Data Availability Statement: Not applicable.

Acknowledgments: This work would not have been possible without the help of my supervisor, Professor Ahmed Alzahrani. I would like to thank him for his expert advice and usual support.

Conflicts of Interest: The authors declare that there are no conflict of interest regarding the publication of this paper.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
DDoS	Distributed Denial of Service
IDS	Intrusion Detection System
EWMA	Exponentially Weighted Moving Average
KNN	K-Nearest Neighbors
CUSUM	Cumulative Sum Algorithm
FPR	False Positive Rate
ML	Machine Learning
DR	Detection Rate
LSTM	Long Short-Term Memory
DoS	Denial of Service

NIDS	Network Intrusion Detection System
MLP	Multilayer Perceptron
ADFA-LD	Australian Defense Force Academy Linux Dataset
ADFA-WD	Australian Defense Force Academy Windows Dataset
UNSW	University of New South Wales
SMOTE	Synthetic Minority Over-sampling Technique
SHAP	SHAPley Additive exPlanations
LR	Logistic Regression
SVM	Support Victor Machine
RF	Random Forest
NB	Naive Bayes
DT	Decision Tree
DNS	Domain Name Server
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
TPR	True Positive Rate
TNR	True Negative Rate
FNR	False Negative Rate
DL	Deep Learning

Appendix A

The figure below presents our calculation of the precision, recall, F1 score and accuracy of our model compared with those of other algorithms.

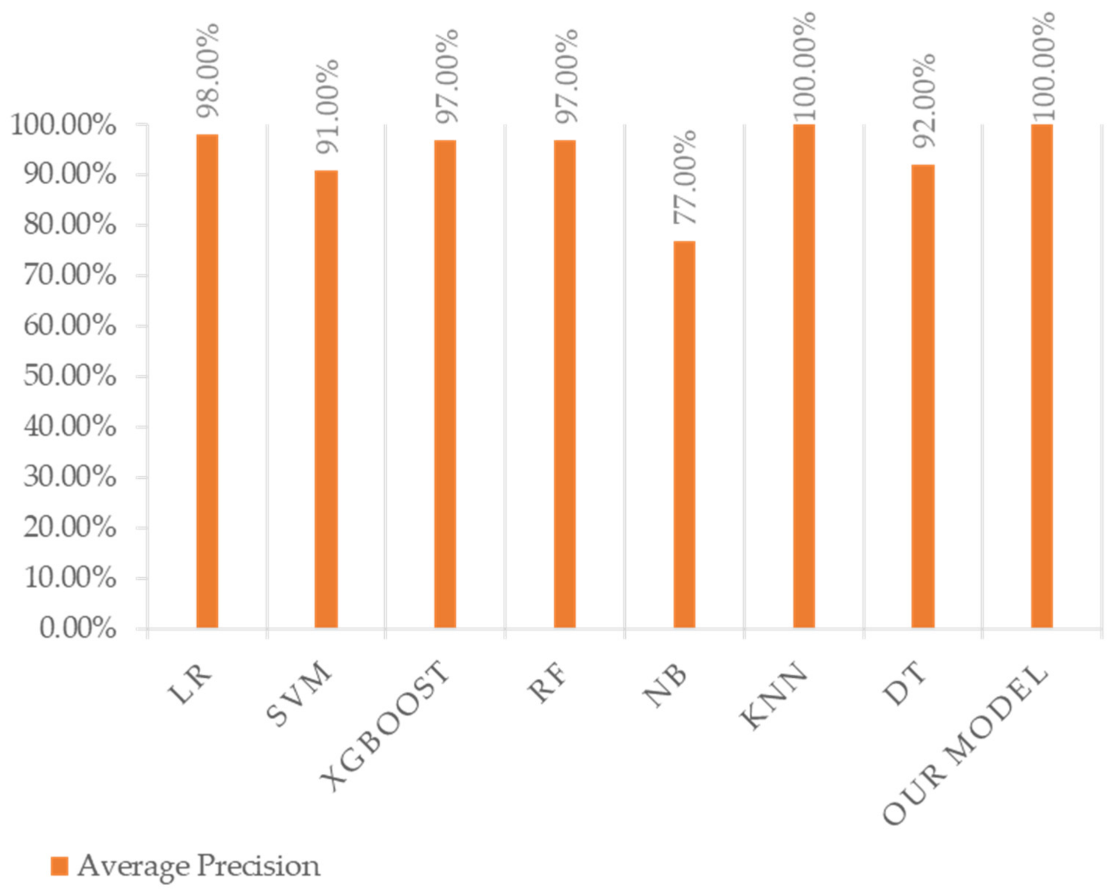


Figure A1. Comparison of the precision of our model with different algorithms.

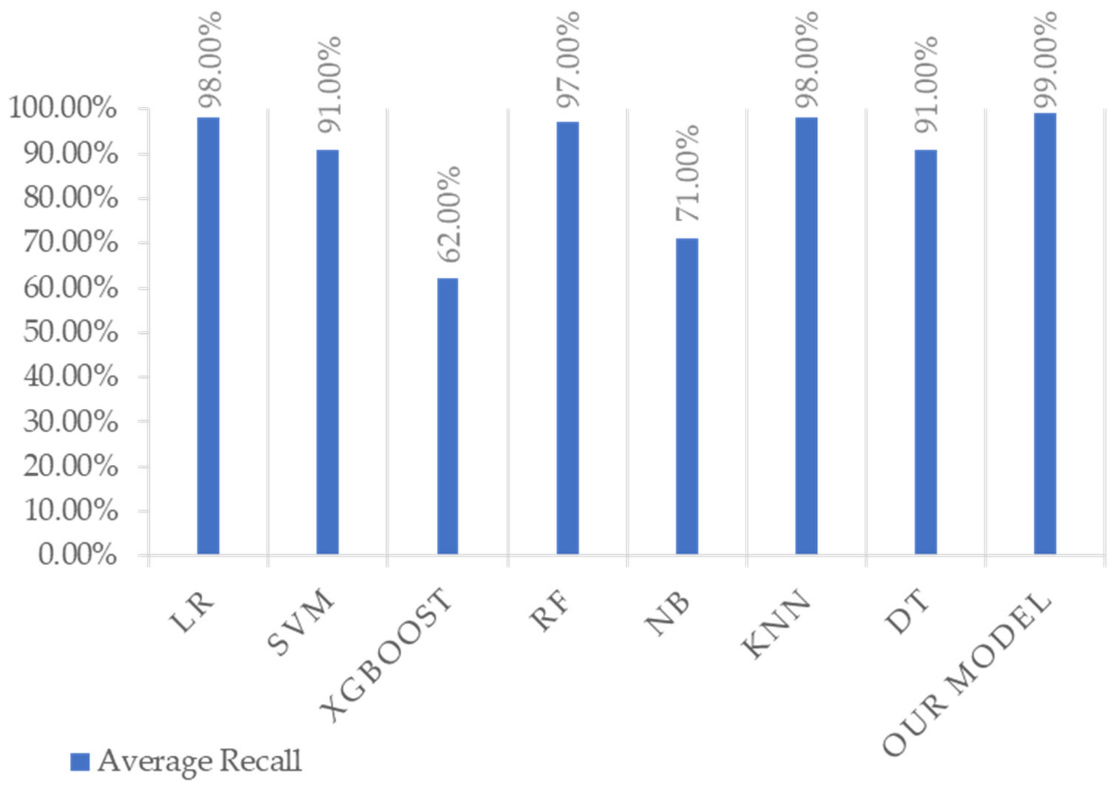


Figure A2. Comparison of the recall of our model with different algorithms.

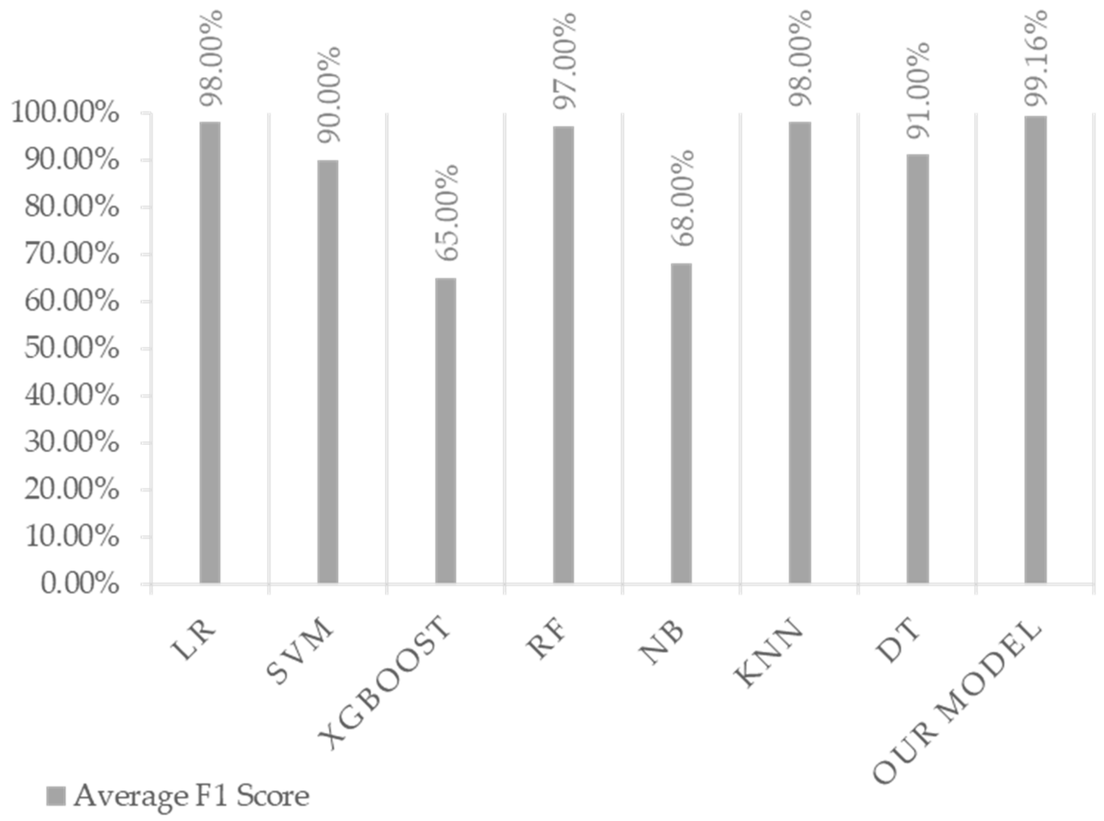


Figure A3. Comparison of the F1 score of our model with different algorithms.

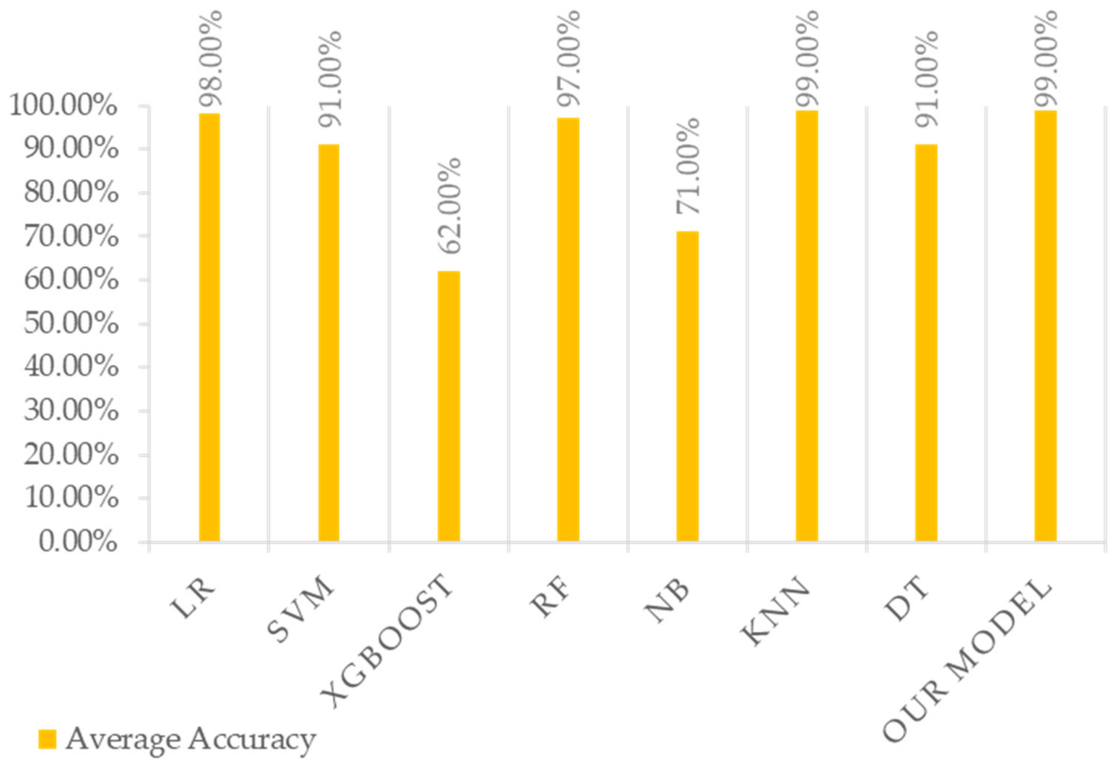


Figure A4. Comparison of the accuracy of our model with different algorithms.

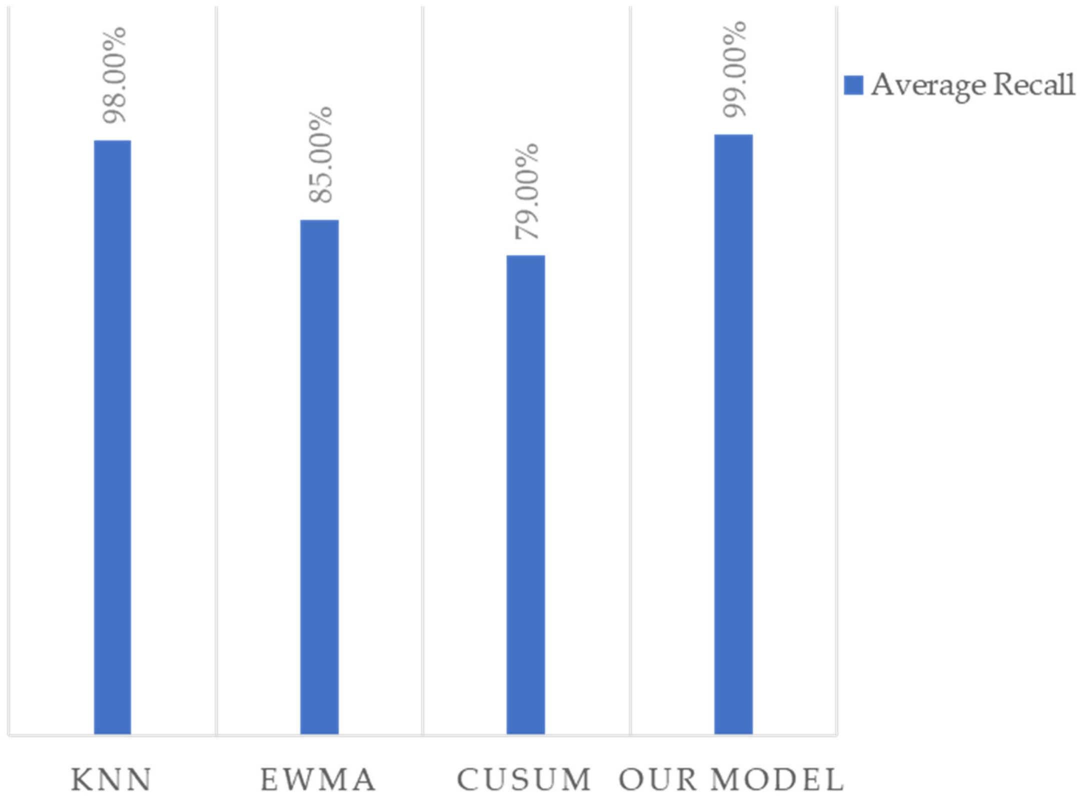


Figure A5. Comparison of the recall of EWMA, CUSUM, and KNN with our model.

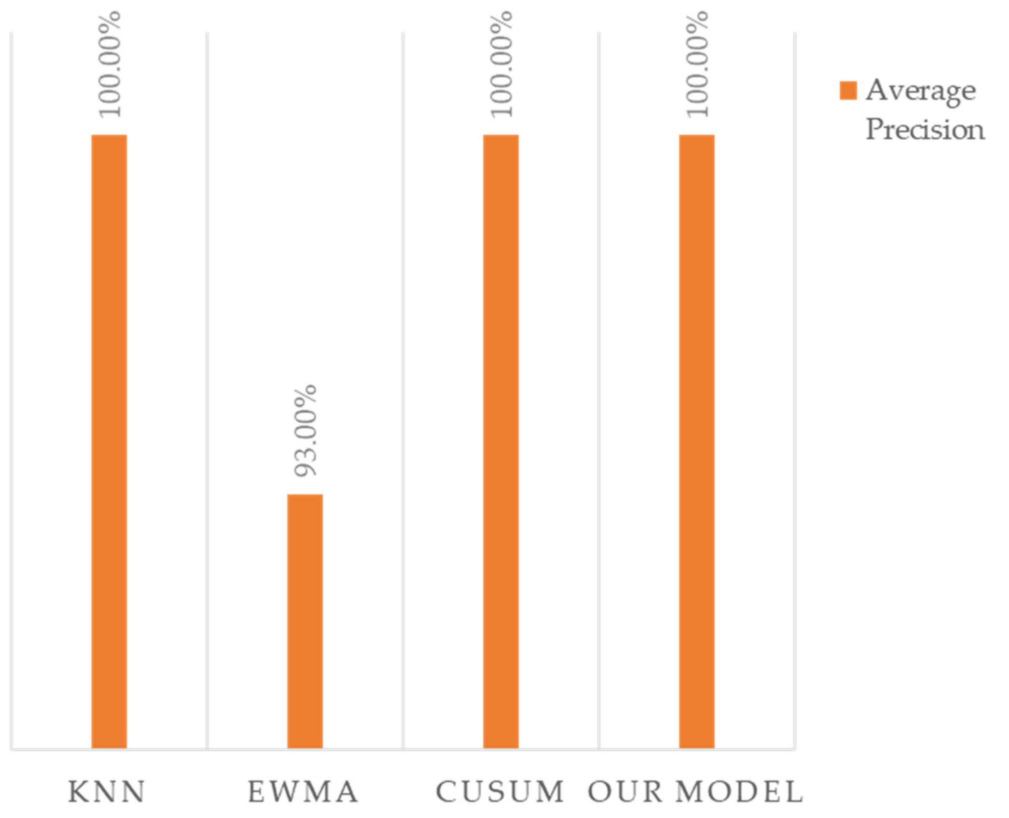


Figure A6. Comparison of the precision of EWMA, CUSUM, and KNN with our model.

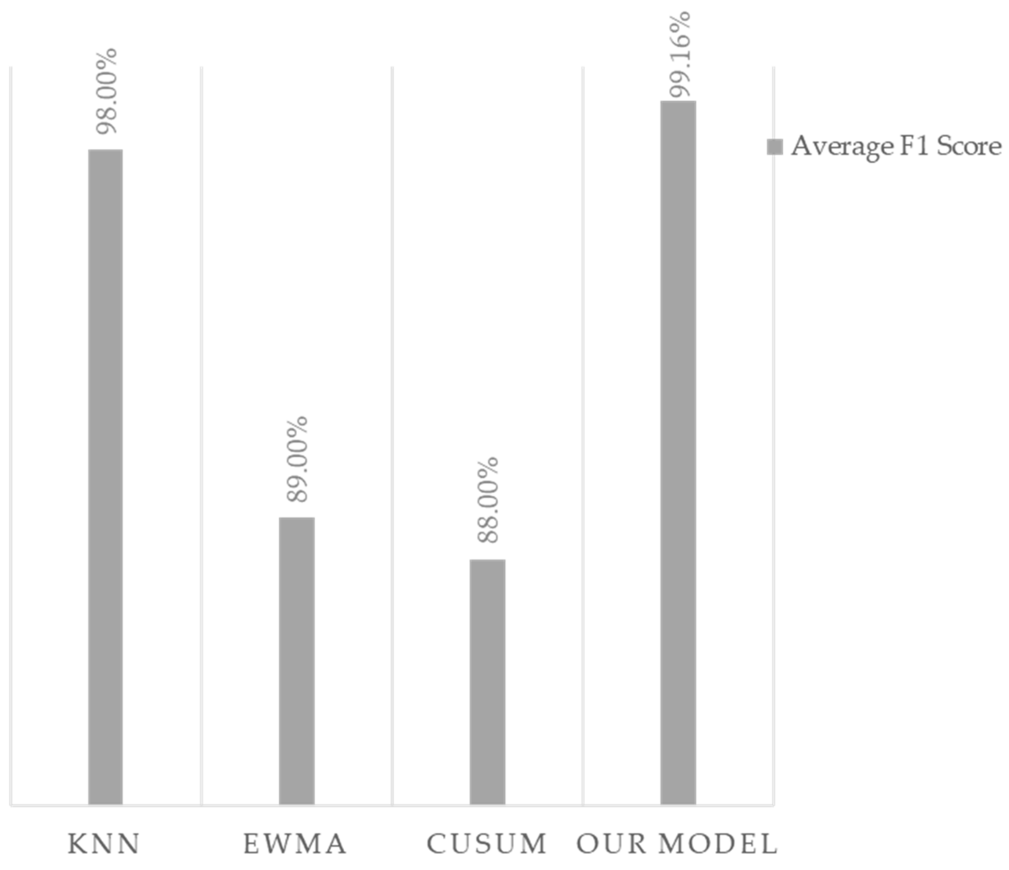


Figure A7. Comparison of the F1 score of EWMA, CUSUM, and KNN with our model.

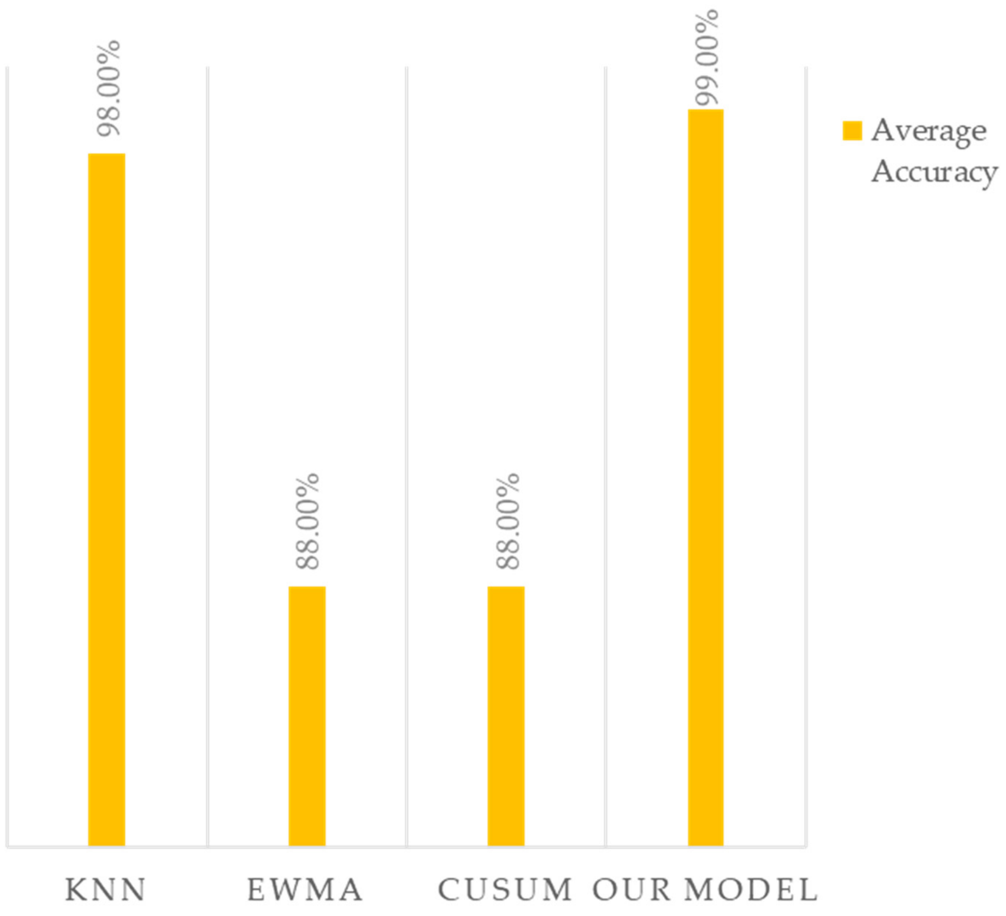


Figure A8. Comparison of the accuracy of EWMA, CUSUM, and KNN with our model.

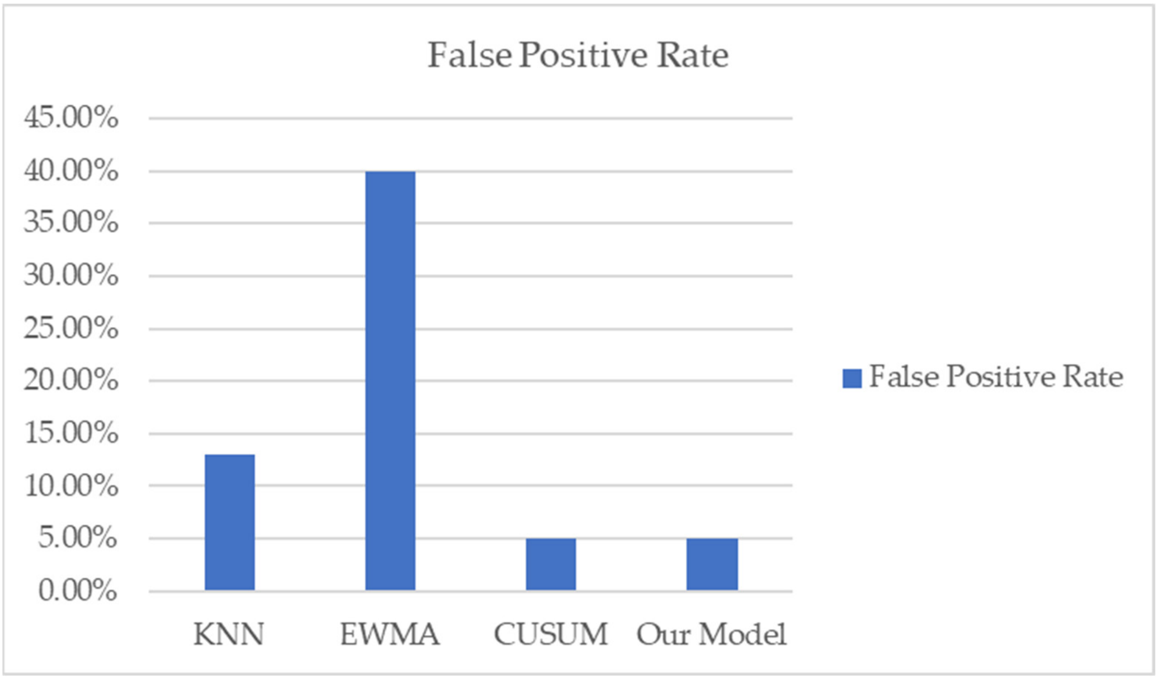


Figure A9. Comparison of the FPR of EWMA, CUSUM, and KNN with our model.

References

1. Ashkan, Y.; Caleb, F.; Tam, N.; Krishna, K.; Fatemeh, J.; Amirreza, N.; Jian KJue Jason, P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *J. Syst. Archit.* **2019**, *98*, 289–330. [CrossRef]
2. Palle, S.R.G. Fog Computing Based IoT Applications and Their Performance. Ph.D. Thesis, University of Missouri-Kansas City, Kansas City, MO, USA, 2018.
3. Zhou, L.; Guo, H.; Deng, G. A fog computing based approach to dDoS mitigation in iIoT systems. *Comput. Secur.* **2019**, *85*, 51–62. [CrossRef]
4. Paharia, B.; Bhushan, K. Fog Computing as a Defensive Approach Against Distributed Denial of Service (dDoS): A Proposed Architecture. In Proceedings of the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 10–12 July 2018. [CrossRef]
5. Alzahrani, R.J.; Alzahrani, A. Security Analysis of dDoS Attacks Using Machine Learning Algorithms in Networks Traffic. *Electronics* **2021**, *10*, 2919. [CrossRef]
6. Rami, J.A.; Ahmed, A. Survey of Traffic Classification Solution in IoT Networks. *Int. J. Comput. Appl.* **2021**, *183*, 37–45.
7. Özçelik, I.; Brooks, R.R. Cusum-entropy: An efficient method for dDoS attack detection. In Proceedings of the 2016 4th International Istanbul Smart Grid Congress and Fair (ICSG), Istanbul, Turkey, 20–21 April 2016; pp. 1–5.
8. Leu, F.Y.; Yang, W.J. Intrusion Detection with CUSUM for TCP-Based dDoS. In Proceedings of the Embedded and Ubiquitous Computing—EUC 2005 Workshops, Nagasaki, Japan, 6–9 December 2005; pp. 1255–1264.
9. Zhang, T. Cumulative sum algorithm for detecting SYN flooding attacks. *arXiv* **2012**, arXiv:1212.5129.
10. Patel, D.; Srinivasan, K.; Chang, C.-Y.; Gupta, T.; Kataria, A. Network Anomaly Detection inside Consumer Networks—A Hybrid Approach. *Electronics* **2020**, *9*, 923. [CrossRef]
11. Ahsan, M.; Mashuri, M.; Kuswanto, H.; Prastyo, D.D. Intrusion Detection System using Multivariate Control Chart Hotelling's T2 based on PCA. *Int. J. Adv. Sci. Eng. Inf. Technol.* **2018**, *8*, 1905–1911. [CrossRef]
12. Sales Mendes, A.; Jiménez-Bravo, D.M.; Navarro-Cáceres, M.; Reis Quietinho Leithardt, V.; Villarrubia González, G. Multi-Agent Approach Using LoRaWAN Devices: An Airport Case Study. *Electronics* **2020**, *9*, 1430. [CrossRef]
13. Meidan, Y.; Bohadana, M.; Shabtai, A.; Guarnizo, J.D.; Ochoa, M.; Tippenhauer, N.O.; Elovici, Y. ProfilIoT: A machine learning approach for IoT device identification based on network traffic analysis. In Proceedings of the ACM Symposium on Applied Computing, Marrakech, Morocco, 3–7 April 2017; pp. 506–509. [CrossRef]
14. Ortiz, J.; Crawford, C.; Le, F. DeviceMien: Network device behavior odellingg for identifying unknown IoT devices. In Proceedings of the IoTDI 2019 Internet of Things Design and Implementation, Montreal, QC, Canada, 15–18 April 2019; pp. 106–117. [CrossRef]
15. Sivanathan, A.; Gharakheili, H.H.; Loi, F.; Radford, A.; Wijenayake, C.; Vishwanath, A.; Sivaraman, V. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. *IEEE Trans. Mob. Comput.* **2019**, *18*, 1745–1759. [CrossRef]
16. Bai, L.; Yao, L.; Kanhere, S.S.; Wang, X.; Yang, Z. Automatic Device Classification from Network Traffic Streams of Internet of Things. In Proceedings of the Conference on Local Computer Networks (LCN), Chicago, IL, USA, 1–4 October 2018; pp. 597–605. [CrossRef]
17. 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*, 100391. [CrossRef]
18. Mihoub, A.; Fredj, O.B.; Cheikhrouhou, O.; Derhab, A.; Krichen, M. Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques. *Comput. Electr. Eng.* **2022**, *98*, 107716. [CrossRef]
19. Chaabouni, N.; Mosbah, M.; Zemmari, A.; Sauvignac, C.; Faruki, P. Network intrusion detection for IoT security based on learning techniques. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2671–2701. [CrossRef]
20. Sudqi Khater, B.; Abdul Wahab AW, B.; Idris MY, I.B.; Abdulla Hussain, M.; Ahmed Ibrahim, A. A lightweight perceptron-based intrusion detection system for fog computing. *Appl. Sci.* **2019**, *9*, 178. [CrossRef]
21. Strecker, S.; Van Haaften, W.; Dave, R. An analysis of IoT cyber security driven by machine learning. In Proceedings of the International Conference on Communication and Computational Technologies: ICCCT 2021, Virtually, 24 August 2021; Springer: Singapore, 2021; pp. 725–753.
22. Sklavounos, D.; Edoh, A.; Plytas, M. A Statistical Approach Based on EWMA and CUSUM Control Charts for R2L Intrusion Detection. In Proceedings of the 2017 Cybersecurity and Cyberforensics Conference (CCC), London, UK, 21–23 November 2017; pp. 25–30.
23. Peterson, J.M.; Leevy, J.L.; Khoshgoftaar, T.M. A Review and Analysis of the Bot-IoT Dataset. In Proceedings of the 2021 IEEE International Conference on Service-Oriented System Engineering (SOSE), Oxford, UK, 23–26 August 2021; pp. 20–27. [CrossRef]
24. Zheng, Y.; Kwoh, C.K. A feature subset selection method based on highdimensional mutual information. *Entropy* **2011**, *13*, 860–901. [CrossRef]
25. Hall, G. Pearson's Correlation Coefficient. 2015. Available online: http://www.hep.ph.ic.ac.uk/~hallg/UG_2015/Pearsons.pdf (accessed on 16 December 2022).
26. Lundberg, S.M.; Erion, G.G.; Lee, S.I. Consistent individualized feature attribution for tree ensembles. *arXiv* **2018**, arXiv:1802.03888.
27. Kumari, K.; Mrunalini, M. Detecting Denial of Service attacks using machine learning algorithms. *J. Big Data* **2022**, *9*, 56. [CrossRef]
28. Kuchimanchi, G.K.; Phoha, V.V.; Balagani, K.S.; Gaddam, S.R. Dimension reduction using feature extraction methods for Real-time misuse detection systems. In Proceedings of the Fifth Annual IEEE SMC Information Assurance Workshop, West Point, NY, USA, 10–11 June 2004; IEEE: Piscataway, NJ, USA, 2004; p. 195202.

29. Machaka, P.; McDonald, A.; Nelwamondo, F.; Bagula, A. Using the cumulative sum algorithm against distributed denial of service attacks in internet of things. In *ICCASA*; Springer: Cham, Switzerland, 2015; pp. 62–72.
30. Cisar, P.; Bošnjak, S.; Cisar, S.M. EWMA algorithm in network practice. *Int. J. Comput. Commun. Control* **2010**, *5*, 160–170. [CrossRef]
31. Atawodi, I.S. A Machine Learning Approach to Network Intrusion Detection System Using K Nearest Neighbor and Random Forest. Master's Thesis, University of Southern Mississippi, Hattiesburg, MS, USA, 2019.
32. Sabireen, H.; Neelanarayanan, V. A review on fog computing: Architecture, fog with IoT, algorithms and research challenges. *Ict Express* **2021**, *7*, 162–176.
33. Singh, V.; Pencina, M.; Einstein, A.J.; Liang, J.X.; Berman, D.S.; Slomka, P. Impact of train/test sample regimen on performance estimate stability of machine learning in cardiovascular imaging. *Sci. Rep.* **2021**, *11*, 14490. [CrossRef] [PubMed]
34. Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Network anomaly detection: Methods, systems and tools. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 303–336. [CrossRef]