Unsupervised Learning for Feature Selection: A Proposed Solution for Botnet Detection in 5G Networks

Moemedi Lefoane, Ibrahim Ghafir, Sohag Kabir, and Irfan-Ullah Awan

***Abstract*—The world has seen exponential growth in deploying Internet of Things (IoT) devices. In recent years, connected IoT devices have surpassed the number of connected non-IoT devices. The number of IoT devices continues to grow and they are becoming a critical component of the national infrastructure. IoT devices' characteristics and inherent limitations make them attractive targets for hackers and cyber criminals. Botnet attack is one of the serious threats on the Internet today. This article proposes pattern-based feature selection methods as part of a machine learning (ML) based botnet detection system. Specifically, two methods are proposed: the first is based on the most dominant pattern feature values and the second is based on Maximal Frequent Itemset (MFI) mining. The proposed feature selection method uses Gini Impurity (GI) and an unsupervised clustering method to select the most influential features automatically. The evaluation results show that the proposed methods have improved the performance of the detection system. The developed system has a True Positive Rate (TPR) of 100% and a False Positive Rate (FPR) of 0% for best performing models. In addition, the proposed methods reduce the computational cost of the system as evidenced by the detection speed of the system.**

***Index Terms*—Botnet Attack, Internet of Things, Network Security, Intrusion Detection System, Machine Learning, Feature Selection.**

I. INTRODUCTION

THE world has seen an influx of IoT devices into the market. Many large organisations have invested in this technology. As a result, countless IoT devices have been developed and continue to be rolled out in the market [1]. The number of IoT devices connected to the internet has been growing in recent years, surpassing the number of connected non-IoT devices; this number grew to 12.3 billion in 2021 [2]. Furthermore, the number of connected IoT devices is projected to reach more than 27 billion IoT connections by 2025 [2]. These devices have now become an integral part of technology playing a crucial role in critical national infrastructures such as smart homes, smart cities, smart grids, health care systems, and intelligent transport [1]. The adoption of IoT technology to the market has opened doors and unlocked so much potential in ways that could not be imagined before, which has sparked interest across several big global organisations, academia and governments.

For some organisations these IoT devices are developed with security in mind; security updates and patches are sent to update devices. However, some vendors do not regularly keep this practice for their devices' life span. It is common for vendors to announce an end of support date for some of the products, unfortunately such an announcement does not lead to affected devices being pulled off the market and continues in operation [1]. For IoT, such devices are deployed in large numbers. This adds to the existing problem, for which experience has shown that many devices remain in operation without regular updates, including security patches. Furthermore, gaps exist between security requirements, current IoT devices and security capabilities. These gaps include limitations inherent to IoT devices, such as processing power capabilities and battery and memory constraints [3].

For low-powered and memory-constrained IoT devices, executing computationally intensive security tasks becomes challenging. Furthermore, the the fifth generation wireless network development is on the rise. These 5G networks have better coverage, higher bandwidth, low latency and low cost. These features will allow 5G to be rapidly deployed in most industries [4]. This makes it highly challenging to deploy complex and robust security measures, thus rendering IoT devices weak, easy targets and low-hanging fruits for adversaries. As such, cybercriminals target and compromise IoT devices for financial gain and criminal activities. Various cyber attacks have existed and continue to evolve, these include Distributed Denial of Service Attacks (DDoS) [5]. DDoS attacks are among the attacks responsible for the loss of billions to organisations across the world and potentially loss of life in the health sector [6], [7]. Another severe attack that has been aimed at IoT devices in recent years is a botnet attack. Botnets are powerful and sophisticated, capable of launching further catastrophic attacks, for example DDoS [8], [9], [10].

The rapid proliferation of IoT will be further accelerated by 5G, thus giving rise to new security challenges, these challenges include DDoS attacks predominantly launched from botnet attacks [4]. Existing detection methods do not consider the future wireless network requirements, such as low latency, which means that these methods may not work for future wireless networks which will form a significant portion of IoT.

IoT challenges and security requirements call for novel and robust techniques to secure IoT devices; the techniques developed must meet the requirements of future wireless technologies (5G), which will form a core component of IoT. Adversaries have been developing novel and sophisticated ways to evade detection techniques. Regardless of sophisti-

TII-21-5691. (Corresponding author: M. Lefoane)

The authors are with University of Bradford, Bradford, United Kingdom (email: mlefoane@bradford.ac.uk; ighafir@bradford.ac.uk; skabir2@bradford.ac.uk; iuawan@bradford.ac.uk

cated techniques adversaries develop, they will always leave a trail that can be analysed to develop state-of-the-art solutions to counter attacks. This leaves network traffic analysis as the best shot at detecting malicious activities as this is where the actual attack occurs.

ML and Intrusion Detection Systems (IDSs) are active areas of research solutions and often yield promising results when securing IoT systems [11]. For ML, one area that continues to produce novel ways that improve performance for detection of maliciousness is the feature selection, feature extraction and feature engineering sub-stage in ML. Within IDSs, ML techniques are deployed to successfully detect network intrusions. ML and IDSs are promising solutions for detecting and mitigating malicious activities because, in these techniques, analysis is performed on network traffic or network traffic-generated data. These techniques can uncover underlying characteristics in network traffic relating to malicious activities. ML and IDSs are therefore best suited for addressing a critical requirement for detecting malicious activities in real-time. It is worth noting that when machine learning algorithms are applied for safety and security applications, it is important to create explainable models such that the reasons behind taking any specific action by the models can be explained. Another important aspect that needs to be considered while applying ML models for security applications is their safety and trustworthiness [12].

This work proposes novel pattern mining feature selection methods based on our previous work [13]. These methods aim to identify the most and least informative features for classification purposes. Once identified, the less informative features are removed, and the most useful ones are used for developing a botnet traffic detection model. Specifically, the two methods proposed are the most dominant pattern frequency-based and MFI mining. The proposed feature selection methods differ from existing filter-based feature selection methods such as chi-square [14] by selecting features based on intrinsic properties of features or underlying patterns without utilising target labels. Whilst existing filter-based feature selection methods rely on the target labels to determine the usefulness score of each of the features (only the automated feature selection relies on the target labels to validate the uncovered patterns), the proposed methods have the potential to uncover characteristics of emerging attacks. Filter based feature selection methods like chi-square work well with categorical data types and may not work well with a continuous variable, the proposed methods work well with both categorical and continuous variables.

This work proposes further an unsupervised clustering method for the automatic selection of most influential features, which addresses the limitation of existing predefined threshold-based feature selection methods. The proposed methods are evaluated using a standard dataset on Logistic Regression (LR), Decision Tree (DT) and Support Vector Machines (SVM) algorithms, which are popular in the literature for addressing similar research problems [15], [16], [17], [18]. The main contributions of this work are summarised as follows:

- Proposed feature selection methods enable the development of a cost-effective model in terms of computational cost, as evidenced by improved network connections detection rate, the detection rate is less than 1ms which meets the requirements for future 5G wireless networks.
- Automated selection of features based on GI score and *K-Means* clustering allows for developing an effective and accurate model that maximises *TPR* and minimises *FPR*. Selection of the best set of features for training ML model is a limitation that exists within filter-based feature selection methods which always rely on a selected threshold to filter out non-informative features.
- The proposed feature selection methods improve the results of the non-performing model (LR) to become best-performing model.

II. Related Works on Botnet and Malware Detection

Several studies investigate deep learning approaches for IoT botnet detection [19]. In Popoola et al. [20]'s work, a Federated Deep Learning method is proposed to address privacy concerns that arise during the detection of zero-day botnet attacks in IoT. The privacy concerns arise from Deep Learning approaches such as centralised Deep Learning. For the proposed approach they deploy Deep Neural Networks (DNN) architecture for the classification of network traffic. The proposed approach [20] is evaluated on a simulated zero-day attack using datasets derived from BoT-IoT and N-BaIoT. The reported performance results indicate the detection of zero-day attacks with high accuracy. In addition, the proposed approach guarantees privacy, has low communication overhead, low memory space for training data storage and low network latency.

Popoola et al. [21] investigated botnet detection approaches using a hybrid deep learning mechanism, specifically, they proposed long short-term memory for encoding and deep Bidirectional long short-term memory (BLSTM). They evaluated the proposed approach on the BoT-IoT dataset and reported a 91.89% reduction in traffic data storage, thereby outperforming the state-of-the-art. The proposed deep learning approach is reported to be robust against overfitting and underfitting.

In their work, Salim et al. [22] studied DDoS attacks and defences in IoT. Furthermore, their study revealed factors and characteristics that make them preferable for carrying out DDoS attacks. These factors are listed as follows: IoT devices are connected continuously, lack sufficient security protocols, if any, weak default password credentials that are rarely changed or updated from default, and lack of capability to reset devices after an attack (resilience). Operating without firmware updates, targeting and using IoT for attacks is cost-effective. Their work classified DDoS attacks into two broad categories namely: bandwidth and resource depletion attacks.

Ali et al. [23] performed a systematic literature review on IoT botnet attacks. Their work revealed that research in IoT botnet attack detection is an active field and has been gaining momentum in the last three years. Securing IoT against botnet attacks is still in early stages and an active field of research that demands further investigation to counter adversaries' ever-evolving landscape of new and novel attack strategies. In recent years this has been one of the most prevalent and prime targets by adversaries.

Wiyono et al. [24] analysed network forensic techniques to detect botnet activities in IoT. Their analysis utilised Decision Tree C4.5. One of the benefits of why this was chosen is that it runs faster and saves power. The proposed approach is evaluated on the BoT-IoT dataset. They reported the following performance for the proposed technique: Accuracy: 97.62%, Precision: 97.63%, Recall: 99.99% and fall-out: 15.7%.

Sudheera et al. [25] proposed a distributed framework for classifying attack stages in botnet attacks. The stages include C&C and scanning stage. The proposed solution is a three-phase approach. In the first phase, the expected behaviour of devices is learned, leading to the generation of standard profiles of devices. Secondly, the monitoring is performed on the traffic. Any deviation from the profile trigger alerts, which are then sent to the manager for further processing. The alerts collected at the manager are analysed, the Frequent Itemset Mining is employed for correlating different types of alerts, leading to the detection of different stages of a botnet attack. The proposed approach is evaluated on publicly available datasets: CTU-IoT-Malware-Capture 34-1, CTU-IoT-Malware-Capture 43-1 and NSS Mirai dataset. The evaluation is performed with three ML algorithms: K-NN, RF and SVM.

Velasco-Mata et al. [16] proposed an optimized machine learning approach to botnet detection in IoT. They proposed two feature selection methods based on information gain and Gini importance. Their proposed approach was evaluated on the CTU-13 dataset. One of the reported challenges is an imbalance in the dataset. Three ML algorithms employed for evaluation are DT, K-NN and RF. They reported the highest performance F-Score of 85%.

Chettri and Bera [26] investigated IoT challenges and vision in 5G, outlined enabling technologies and proposed an IoT architecture in 5G. The study highlighted the limitations of existing 3G and 4G networks that cannot be utilised for IoT wireless networks such as Low Power Wide-Area networks. On the other hand, the IoT wireless networks can access 5G network, thus making 5G a game changer for IoT. Ghorbani et al. [4] investigate DDoS attacks on IoT in the advent of 5G, they argue that the combination of IoT and 5G give rise to new security challenges that exploit both technologies' architecture vulnerabilities. Lucas-Estañ and Gozalvez studied and proposed Grant-Free Scheduling for reliable low latency in 5G networks. The proposed approach reduces the transmission delay, affecting latency, a crucial requirement for 5G [27]. Reducing latency requirements for some of the tasks in 5G means that existing IoT botnet detection methods may be rendered ineffective if their processing time is more than the scheduling allocated time.

III. Proposed Methodology

Fig. 1 shows the proposed deployment scenario for the IoT botnet detection system. The IoT system network is connected to the cloud where IoT devices' data is periodically sent to the cloud, for further analysis. A PCAP (Packet Capture) Logs Capture module generates and stores PCAP and Logs to further develop a real-time model for IoT botnet detection. Zeek[28] is used within the PCAP Logs Capture module. Alternatively, Snort[29] can be used. Network Traffic analysis modelling studies PCAP and Logs. ML models are then trained for the detection of botnet traffic, as explained in Sections III-A and III-B. The detection models developed are deployed on the Botnet Detection Module to monitor the IoT network in real-time. Once an attack is detected in IoT network traffic, an alert is sent to the Security Manager to perform forensic analysis and take appropriate actions to secure affected devices.

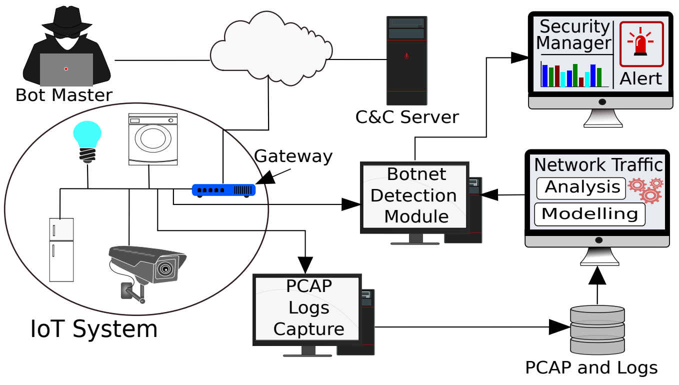


Fig. 1: Proposed topology scenario for IoT botnet detection

The proposed botnet detection system is divided into two stages, namely the Feature Selection stage and the Building the Detection Model stage.

A. Feature Selection

As stated in Section I, this work proposes two feature selection methods. The first method uses a predefined threshold, and the second uses unsupervised learning to select the most influential features. This sub-section provides more details on the proposed feature selection methods.

1) Feature Selection with a predefined threshold: Fig. 2 illustrates the feature selection method with a predefined threshold. The method requires network connections data (All Features Data) and a threshold as input. Determining good features is based on data mining technique FIM, which uncovers underlying patterns from transactions data, FIM captures homogeneity and regularity in the data translating to either valuable or noisy features [30]. From the transactions data, FIM extracts frequently occurring transactions with their minimum support, which represents frequency of occurrence of a pattern in transactions. For this work a network connection represents a transaction, from which FIM patterns are mined to identify useful features. This work utilises an open source data mining java based SPMF to mine patterns [31]. From the connections data, the most dominant pattern values for each feature are computed, and the pattern frequency value of the most dominant pattern is calculated with respect to the total number of connections. This pattern frequency value is compared with a predefined threshold, a cut-off value that determines which features are added to the noisy features list. The noisy features list is then passed to the Noisy Feature

Filter which removes noisy feature names identified from a list of all feature names generated from data. The result is an influential feature list(containing only important) used to derive reduced network traffic data containing only the most influential features.

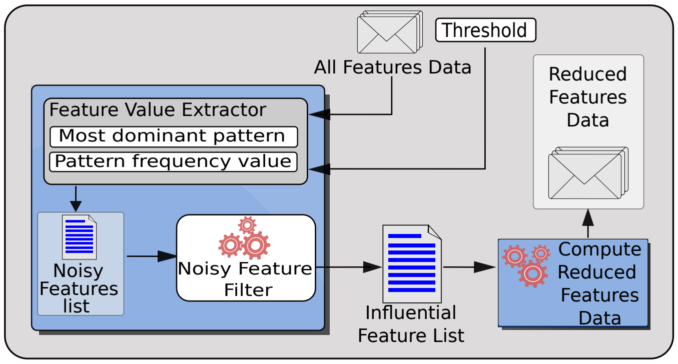


Fig. 2: Proposed feature selection method for IoT botnet detection with a predefined threshold

Algorithm 1 Implementation pseudocode for feature selection method for botnet detection with a predefined threshold

```
Require: $data$                                  ▷ Dataset
Require: $sp$                  ▷ Source port provided by user
Require: $tgt$                     ▷ target provided by user
Require: $thld$                  ▷ frequency of feature value
 1: $r \leftarrow compute\ total\ instances\ (rows)$
 2: $col\_dl[] \leftarrow$ “”   ▷ Create empty list for feature names to drop
 3: $drop\ sp$
 4: $drop\ tgt$
 5: $drop\ id\_col$
 6: for $col\_name$ in $columns$ do
 7:     $v[] \leftarrow compute\ total\ count\ for\ each\ unique\ value$
 8:     for $uniq\_value$ in $col\_name$ do
 9:         if $v[uniq\_value]/r > thld$ then
10:             if $col\_name$ not in $col\_dl$ then
11:                 $col\_dl[] \leftarrow Add\ col\_name\ to\ drop\ list$
12:             end if
13:         end if
14:     end for
15: end for
```

Algorithm 1 provides detailed implementation pseudocode followed for identifying and removing less informative features. This requires source port, target name and a threshold value ranging from 0 to 100. These are removed from the data before employing any feature selection method. The removal of these features is informed by expert knowledge about some of the network traffic features known to be noisy(non-informative). For instance, the source port is non-informative as it is generated randomly. Therefore it does not have any meaningful contribution to classification. In lines 1 and 2, total observations in data are computed and saved in r and an empty list (col_dl) is created to store names of non-informative features. All non-informative features are removed as indicated in lines $3-5$ (based on expert knowledge). Lines $6-11$ identify less influential features (based on the threshold provided). This is done by iterating through each feature, whereby pattern frequency value counts are computed. Furthermore, if any pattern frequency value is greater than the threshold, the feature name is added to the less influential features list. Finally, all the features identified as less informative are removed from the data, resulting in reduced features data passed to Building the Detection Model stage explained in Section III-B.

2) Feature Selection without a predefined threshold: Fig. 3 shows the automatic feature selection stage without a predefined threshold. Firstly, network connections data is fed into *Feature Sample Extractor,* where there are two modules: *Frequency Based (FB)* and *Pattern Based (PB)*. The *FB* module extracts dataset sample by iterating through each feature value and extracting the most dominant pattern frequencies (including features' co-occurrence frequencies) which are then captured in the Features values file. This implies that if a group of features are informative and co-occurs in a pattern, then MFI calculates their pattern frequencies simultaneously. As a result, the frequency score representing informativeness is performed faster with MFI as the frequencies of co-occurring features are not computed individually for each feature.

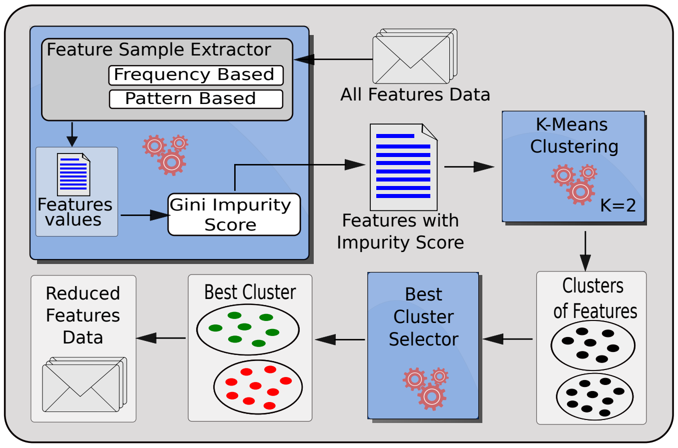


Fig. 3: Proposed feature selection method for IoT botnet detection without a predefined threshold

Most dominant patterns identified for each feature need to be validated to determine the usefulness of patterns in detecting botnet traffic. A GI score for each feature is proposed to determine its informativeness, which is used to measure its quality. A most dominant feature's GI score is considered pure if its most dominant pattern value is only present in one target class (either Normal or Malicious). GI is a technique that has been used in the DT classification algorithm as a splitting criterion [32]. GI is often chosen in DT compared to the alternatives, entropy or information gain, which are more computationally expensive, hence GI being chosen as an efficient method. This work uses GI not as a splitting criterion but rather as the measure of usefulness of features based on a pattern obtained from a sample of network traffic data. The output is features with their impurity scores captured in a file (Features with Impurity Score).

The GI score captures the level of impurity. The GI scores of each of the features are used to select the best set of features the same way chi-square is used to select best feature for categorical features by picking only the most pure features and discarding impure ones to maximise performance. On the other hand, if the most dominant feature pattern is present in all target classes, it is considered to have impurities. The formula for computing GI is given by equation (1), where D is the dataset, k is the number of classes contained in the dataset and p_c is the probability that a random sample belongs to class c.

$$Gini(D) = 1 - \sum_{c=1}^{k} p_c^2 \tag{1}$$

K-Means clustering module automatically selects the best set/cluster of features for the Building the Detection Model stage. *K-Means* clustering module uses Features with Impurity Score to cluster features. The assumption is that features are either informative or noisy. The expectation is that the most influential features are clustered together, and similarly noisy features are clustered together in the same cluster. As a result, two centroids (of the most pure feature and the least pure feature) further apart are used to vote on features closest to each other. The number of target classes (Malicious and Normal) informs selecting $k = 2$ for *K-Means* clustering, i.e. most influential features are clustered together and likewise noisy features clustered together.

Illustrated in Algorithm 2 is implementation pseudocode that shows detailed steps of performing feature selection without a predefined threshold. It takes in network connections data and computes the most dominant pattern value of each feature. Once each feature's most dominant pattern is found, a sample is drawn from network traffic data with the corresponding pattern value. The GI score is computed and stored from this sample for each feature. The output of this module is features with their corresponding GI score, which are passed to the *K-Means* clustering module.

The goal of automatically selecting the best features is achieved in two-steps clustering phases. In the first phase, *K-Means* clustering is deployed with $K = 2$. By clustering features into two clusters according to their GI scores, the unsupervised clustering method effectively splits the features into most influential and less influential clusters. Furthermore, while clustering, the lowest score computed from the GI scores is captured. The output of the first clustering phase results in two clusters, Cluster_A and Cluster_B and the lowest GI score. The second clustering phase determines the best cluster. The best cluster is computed by iterating through Cluster_A to check if it contains the lowest GI score. If it contains the lowest score, Cluster_A is returned as the best cluster and is used to derive reduced feature data. Otherwise Cluster_B is returned as the best cluster. The best cluster is then used to derive reduced feature data that is passed to the next stage.

Algorithm 2 Implementation pseudocode for feature selection method for botnet detection without a predefined threshold

```
Require: $data$                                  ▷ Dataset
Require: $sp$                   ▷ Source port provided by user
Require: $tgt$                       ▷ target provided by user
Require: $id\_col$                     ▷ unique ideantifier ID
 1: $Output \leftarrow \{\}$   ▷ Empty list of lists for feature dominant
    patterns values and score
 2: $drop\ sp$
 3: $drop\ tgt$
 4: $drop\ id\_col$
 5: for $feature\_name$ in $columns$ do
 6:     $max\_count \leftarrow 0$
 7:     $feature\_v \leftarrow 0$
 8:     $feature\_cnts \leftarrow \{\} total\ uniq\ value\ counts$
 9:     for $uniq\_v$ in $feature\_cnts$ do
10:         $v\_count \leftarrow feature\_cnts.get\_count(uniq\_v)$
11:         if $v\_count > max\_count$ then
12:             $max\_count \leftarrow v\_count$
13:             $feature\_v \leftarrow uniq\_v$
14:         end if
15:     end for
16:     $Data_s \leftarrow data(feature\_name = feature\_v)$
17:     $gini(Data_s) \leftarrow 1 - \sum_{c=1}^{k} p_c^2$
18:     $Output.append([feature\_name, feature\_v, gini])$
19: end for
20: Return $Output$
```

B. Building the Detection Model

Fig. 4 shows the Detection Model stage of the proposed method. Here the input is the reduced features dataset from the previous stage. First data goes through Model Training, where three machine learning algorithms are trained namely: LR, DT and SVM. The trained models are then passed to the Model Evaluation module. Different trained models are evaluated in this module, the output is results with evaluation metrics outlined in sub section IV-B for the performance of models trained. The evaluation results are passed to the Best Model Selector, which goes through the evaluation results and selects the best performing model(s). For comparison purposes, the original dataset is also provided as an input into this stage for evaluating the performance of the proposed methods.

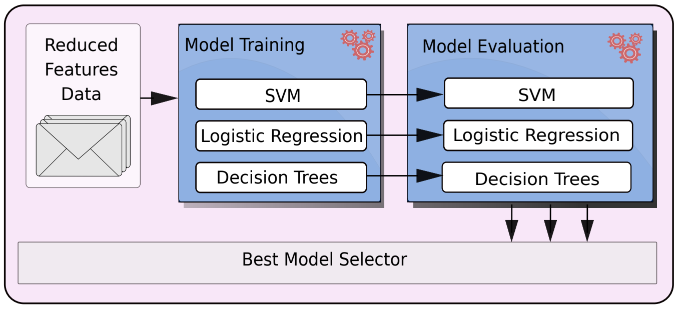


Fig. 4: Proposed Building the Detection Model Stage for IoT Botnet Detection.

IV. Evaluation of Results

To evaluate the effectiveness of the proposed methods, a publicly available dataset IoT23 [33] is used. Since this work aims to detect botnet traffic from the network traffic connections data, this makes it a binary classification problem. For this reason, all malicious labels for the dataset namely: C&C, DDoS and PartOfHorizontalScan, were combined into one class. This resulted in two classes: normal and malicious. Experiments are performed in two scenarios–the first scenario, experiment is performed without the proposed feature selection methods whereas the second scenario uses the proposed feature selection methods for feature selection presented in Figs. 2 and 3 respectively. The results of both scenarios are used to build and evaluate the detection model as presented in Fig. 4.

A. Dataset Description

IoT23 is a dataset derived from IoT traffic. Refer to Table I for label distribution of the chosen dataset. This dataset is labelled connection log files from the network traffic generated by Zeek [28]. The dataset has 20 features, namely: ts, uid, id_orig_h, id_orig_p, id_resp_h, id_resp_p, proto, service, duration, orig_bytes, resp_bytes, conn_state, local_orig, local_resp, missed_bytes, history, orig_pkts, orig_ip_bytes, resp_pkts, and resp_ip_bytes. The classification of these features is presented in Table I. The features selected by the proposed automatic feature selection method include: id_resp_h, id_resp_p, proto, history, conn_state, orig_pkts, and orig_ip_bytes.

TABLE I: CTU-IoT-Malware-Capture-34-1 (Mirai) Labels Distribution

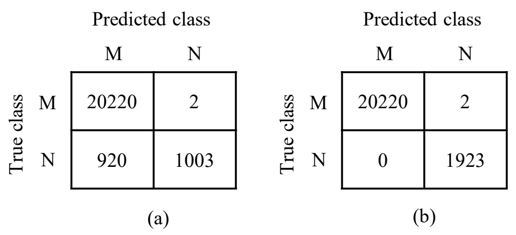
Label	Flows
Normal	1923
C & C	6706
DDoS	14394
PartOfHorizontalPortScan	122

B. Experimental Setup and Results

The metrics computed to evaluate the performance of the proposed methodology are discussed in this sub-section. These metrics provide evidence of how well an IDS makes detection correctly [11]. These metrics are:

- Overall Success Rate (OSR) - the proportion of all correctly detected normal and malicious connections to total connections.
- TPR - the proportion of correctly detected malicious connections to total malicious connections.
- FPR - a measure of the proportion of false alarms to total normal connections.
- $Precision$ - the proportion of correct malicious connections to a total detected as malicious.
- *F-score* - a measure of the trade-off between $Precision$ and TPR. Used for comparison of different classification methods [11].

The proposed method reduced the number of features from twenty to seven. Fig. 5 (a) and Fig. 5 (b) show the confusion matrix for LR NFS and WFS, respectively. NFS-No Feature Selection is applied, refers to the experimental scenario where


(a)

True class \ Predicted class	M	N
M	20220	2
N	920	1003

(b)

True class \ Predicted class	M	N
M	20220	2
N	0	1923

Fig. 5: Confusion Matrix for Logistic Regression (a) NFS (b)WFS

proposed feature selection method is not applied. WFS-With Feature Selection, is a scenario where the proposed method is applied fully as detailed in Section III-A2. The results show consistent improvement across several metrics used. The main benefit of the proposed feature selection method is the reduction of the false alarms indicated by FPR.

Fig. 6 (a) and 6 (b) present the ROC curve for the performance of LR NFS and WFS, Fig. 6 (c) and 6 (d) show the ROC curve for SVM NFS and WFS and finally, Fig. 6 (e) and 6 (f) show the ROC curve for DT NFS and WFS. The results show performance improvement with the proposed feature selection methods. Fig. 7 shows the effect of different threshold values on performance, these were selected randomly to show the problem that could arise from selecting threshold that may not be optimal. As can be seen, due to the selection of different threshold values, the performance can vary significantly.

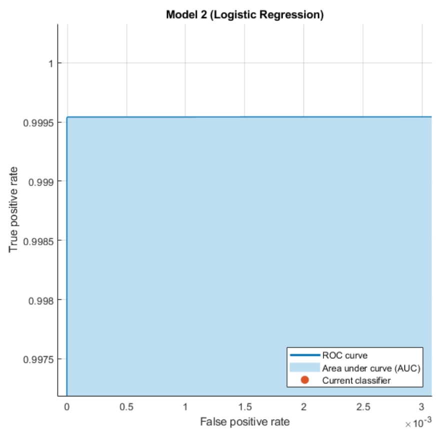
For this reason, the method that requires a predefined threshold may not give optimal performance if the chosen threshold value is not optimal. The problem associated with manually selecting an optimal threshold value is alleviated by the proposed automatic feature selection method without a predefined threshold (c.f. Fig. 3). This method always finds the best features without any human intervention and thus will not suffer from the performance issue as shown in Fig. 7. Fig. 8 shows *FPR* and TPR, $Precision$, OSR, *F-score* as a comparison of performance between methods with and without feature selection.

Fig. 9 compares $Detection\ Speed$ for with and without feature selection methods. The proposed results consistently outperform those obtained without the proposed feature selection methods for all the classification models explored. The resulting speed adheres to the state-of-the-art scheduling requirements, which affects latency in 5G [27]. This makes the proposed method suitable for future wireless networks. Fig. 10 shows the effect of proposed method (WFS) *FPR* compared to NFS. The results consistently reduce false alarms with the proposed method across all models.

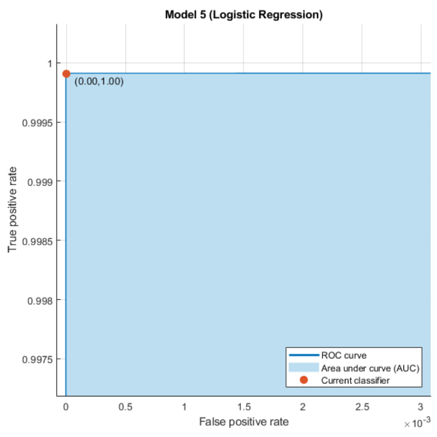
Furthermore, Table II compares methods that utilise the IoT23 dataset. To enable reproducibility of this work, the developed code is made available in [36]. This will be beneficial to stimulate further research interests in the research community.

V. Conclusion

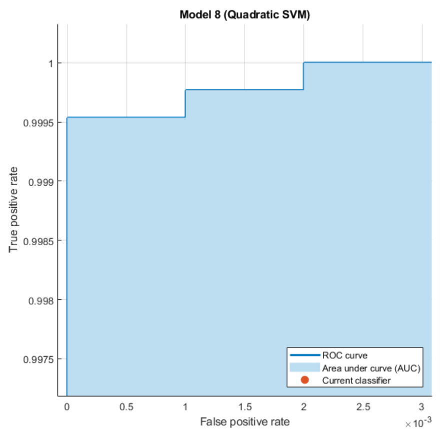
This paper presented an efficient feature selection method for botnet detection. For feature selection, methods with



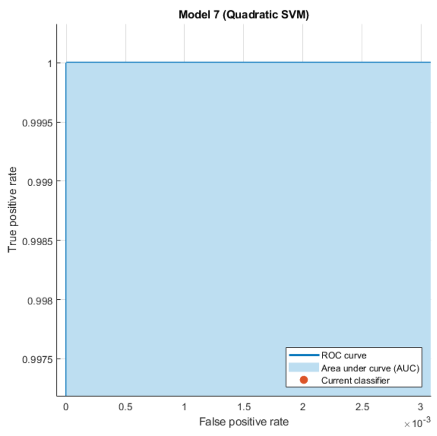
(a) LR NFS



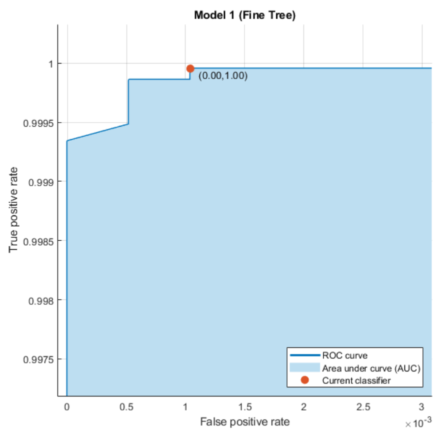
(b) LR WFS



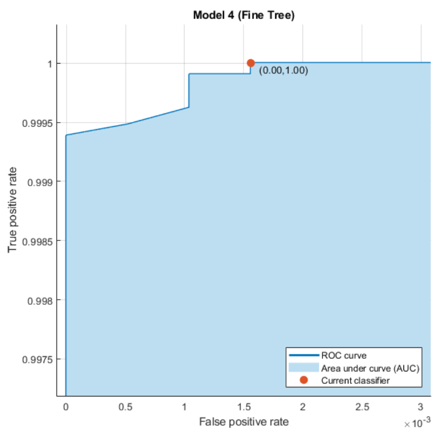
(c) SVM NFS



(d) SVM WFS



(e) DT NFS



(f) DT WFS

Fig. 6: ROC Plots for methods WFS and NFS Performance.

TABLE II: Comparison of Works utilising IoT23 Dataset.

Paper	Problem	Solution	Accuracy	Precision	TPR	F-Score
[25]	Botnet Stage Detection	ML for Stage Detection	99%	X	X	99%
[34]	Botnet Detection	Ensemble ML	99.88%	X	99.7%	X
[35]	Botnet Identification	ML	99.9%	X	X	X
Proposed method	Botnet Identification	ML	99.99%	100%	99.99%	100%

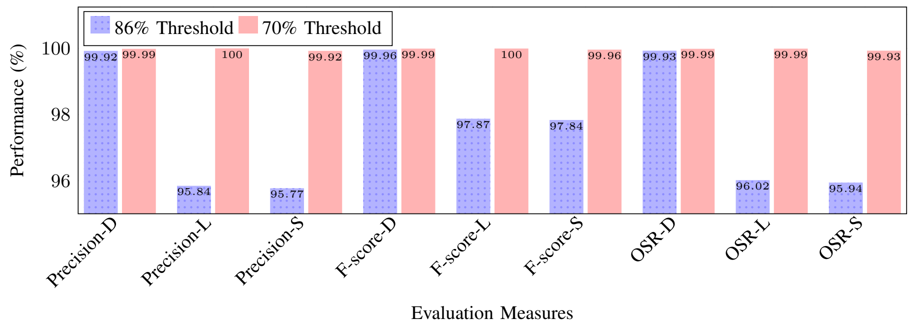


Fig. 7: Comparison of performance between approaches with different feature selection threshold

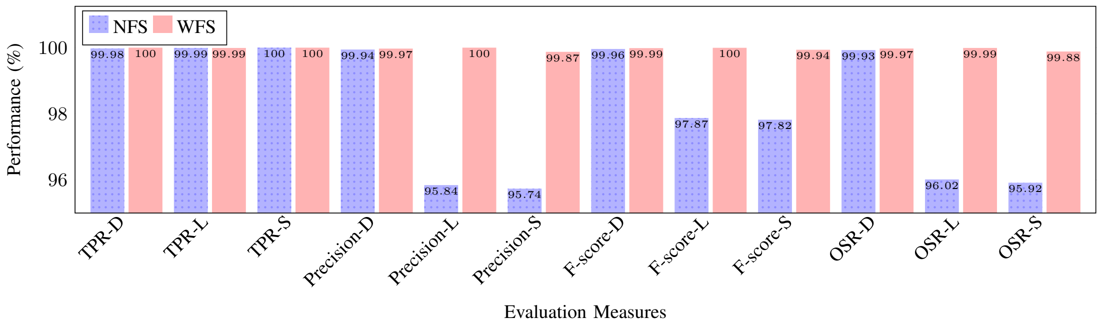


Fig. 8: Comparison of performance between approaches with and without feature selection. D: Decision tree, L: Logistic regression, S: SVM, NFS: No Feature Selection, WFS: With Feature Selection

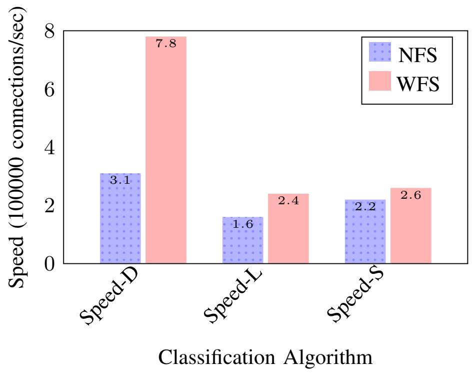


Fig. 9: Comparison of Detection speed between approaches with and without feature selection

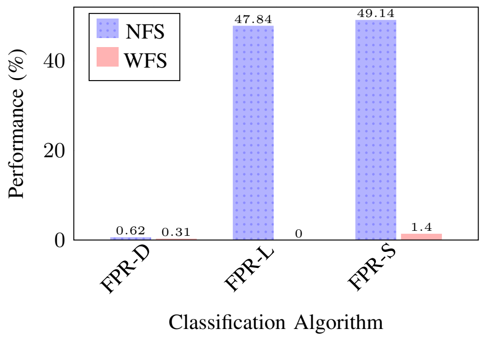


Fig. 10: Comparison of FPR performance between approaches with and without feature selection

and without predefined thresholds are proposed. The method that does not rely on a predefined threshold addresses the problem of selecting an optimal threshold by automatically performing feature selection using *K-Means* clustering and GI score. After feature selection, data with reduced features are processed further in the training and evaluation stage. Specifically, DT, LR and SVM are trained as detection models for botnet traffic detection. Finally, the detection performance of the trained models is evaluated. The results show that the proposed method consistently improves the performance across all the measures: *TPR*, *FPR*, *Precision*, *F-score* and *OSR*. The consistent increase in performance, particularly in *FPR,* suggests that the features selected for training a model identify the best set of features that separate instances well according to malicious and normal instances. Therefore, the selected set of features results in an improved performance that minimises *FPR*. For example, in logistic regression, *FPR* is minimised to zero.

Additionally, the proposed method is efficient and reduces the computational cost regarding detection speed. This is an essential requirement in intrusion detection systems as they must be deployed in real-time. For DT, the results observed from WFS and NFS are closely matched, suggesting the model performs well and does not benefit from the proposed methods. However, if detection speed is considered even DT benefits from the proposed approach. Also with the proposed methods, LR becomes the best-performing model. Furthermore, DT also benefits from the proposed approach and FPR is minimised by half, saving time and network resources. The results show that the proposed system has a very low rate of false alarms.

This protects the network security team from being disturbed by false alarms that might affect their efficiency and waste resources.

References

[1] E. Anthi, L. Williams, M. Słowińska, G. Theodorakopoulos, and P. Burnap, "A supervised intrusion detection system for smart home iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042–9053, 2019.

[2] S. Sinha, "State of iot 2021: Number of connected iot devices growing 9% to 12.3 billion globally, cellular iot now surpassing 2 billion," https://iot-analytics.com/number-connected-iot-devices//, Sep. 2021, accessed: 2021-12-23.

[3] B. K. Mohanta, D. Jena, S. Ramasubbareddy, M. Daneshmand, and A. H. Gandomi, "Addressing security and privacy issues of iot using blockchain technology," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 881–888, 2021.

[4] H. Ghorbani, M. S. Mohammadzadeh, and M. H. Ahmadzadegan, "Ddos attacks on the iot network with the emergence of 5g," in *International Conference on Technology and Entrepreneurship-Virtual (ICTE-V)*. IEEE, 2020, pp. 1–5.

[5] G. Kambourakis, C. Kolias, and A. Stavrou, "The mirai botnet and the iot zombie armies," in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, 2017, pp. 267–272.

[6] U. Raza, J. Lomax, I. Ghafir, R. Kharel, and B. Whiteside, "An iot and business processes based approach for the monitoring and control of high value-added manufacturing processes," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, ser. ICFNDS '17. New York, NY, USA: Association for Computing Machinery, 2017.

[7] M. Hammoudeh, I. Ghafir, A. Bounceur, and T. Rawlinson, "Continuous monitoring in mission-critical applications using the internet of things and blockchain," in *Proceedings of the 3rd International Conference on Future Networks and Distributed Systems*, ser. ICFNDS '19. New York, NY, USA: Association for Computing Machinery, 2019.

[8] D. M. Diab, B. AsSadhan, H. Binsalleeh, S. Lambotharan, K. G. Kyriakopoulos, and I. Ghafir, "Denial of service detection using dynamic time warping," *International Journal of Network Management*, p. e2159, 2021.

[9] I. Ghafir, V. Prenosil, M. Hammoudeh, F. J. Aparicio-Navarro, K. Rabie, and A. Jabban, "Disguised executable files in spear-phishing emails: Detecting the point of entry in advanced persistent threat," in *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems*, ser. ICFNDS '18. New York, NY, USA: Association for Computing Machinery, 2018.

[10] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim, and J. N. Kim, "An in-depth analysis of the mirai botnet," in *2017 International Conference on Software Security and Assurance (ICSSA)*, 2017, pp. 6–12.

[11] I. Ghafir, K. G. Kyriakopoulos, F. J. Aparicio-Navarro, S. Lambotharan, B. Assadhan, and H. Binsalleeh, "A basic probability assignment methodology for unsupervised wireless intrusion detection," *IEEE Access*, vol. 6, pp. 40 008–40 023, 2018.

[12] K. Aslansefat, S. Kabir, A. Abdullatif, V. Vasudevan, and Y. Papadopoulos, "Toward improving confidence in autonomous vehicle software: A study on traffic sign recognition systems," *Computer*, vol. 54, no. 8, pp. 66–76, 2021.

[13] M. Lefoane, I. Ghafir, S. Kabir, and I.-U. Awan, "Machine learning for botnet detection: An optimized feature selection approach," in *The 5th International Conference on Future Networks & Distributed Systems*, ser. ICFNDS 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 195–200.

[14] M. Dhalaria and E. Gandotra, "Android malware detection using chi-square feature selection and ensemble learning method," in *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2020, pp. 36–41.

[15] S. Joshi and E. Abdelfattah, "Efficiency of different machine learning algorithms on the multivariate classification of iot botnet attacks," in *2020 11th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, 2020, pp. 0517–0521.

[16] J. Velasco-Mata, V. González-Castro, E. F. Fernández, and E. Alegre, "Efficient detection of botnet traffic by features selection and decision trees," *IEEE Access*, vol. 9, pp. 120 567–120 579, 2021.

[17] L. Suhuan and H. Xiaojun, "Android malware detection based on logistic regression and xgboost," in *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, 2019, pp. 528–532.

[18] R. Bapat, A. Mandya, X. Liu, B. Abraham, D. E. Brown, H. Kang, and M. Veeraraghavan, "Identifying malicious botnet traffic using logistic regression," in *2018 Systems and Information Engineering Design Symposium (SIEDS)*, 2018, pp. 266–271.

[19] Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, "Machine learning-based iot-botnet attack detection with sequential architecture," *Sensors*, vol. 20, no. 16, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4372

[20] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in iot-edge devices," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3930–3944, 2022.

[21] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the internet-of-things networks," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4944–4956, 2021.

[22] M. M. Salim, S. Rathore, and J. H. Park, "Distributed denial of service attacks and its defenses in iot: a survey," *The Journal of Supercomputing*, vol. 76, pp. 5320–5363, 2019.

[23] I. Ali, A. I. A. Ahmed, A. Almogren, M. A. Raza, S. A. Shah, A. Khan, and A. Gani, "Systematic literature review on iot-based botnet attack," *IEEE Access*, vol. 8, pp. 212 220–212 232, 2020.

[24] R. T. Wiyono and N. D. W. Cahyani, "Performance analysis of decision tree c4.5 as a classification technique to conduct network forensics for botnet activities in internet of things," in *2020 International Conference on Data Science and Its Applications (ICoDSA)*, 2020, pp. 1–5.

[25] K. L. K. Sudheera, D. M. Divakaran, R. P. Singh, and M. Gurusamy, "Adept: Detection and identification of correlated attack stages in iot networks," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6591–6607, 2021.

[26] L. Chettri and R. Bera, "A comprehensive survey on internet of things (iot) toward 5g wireless systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16–32, 2020.

[27] M. C. Lucas-Estañ; and J. Gozalvez, "Sensing-based grant-free scheduling for ultra reliable low latency and deterministic beyond 5g networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4171–4183, 2022.

[28] The-Zeek-Project, "Zeek," https://zeek.org/, 2021, accessed: 2021-10-20.

[29] Cisco, "Snort," https://www.snort.org/, 2021, accessed: 2021-10-20.

[30] J. M. Luna, F. Padillo, M. Pechenizkiy, and S. Ventura, "Apriori versions based on mapreduce for mining frequent patterns on big data," *IEEE Transactions on Cybernetics*, vol. 48, no. 10, pp. 2851–2865, 2018.

[31] P. Fournier-Viger, J. C.-W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, and H. T. Lam, "The spmf open-source data mining library version 2," in *Machine Learning and Knowledge Discovery in Databases*, B. Berendt, B. Bringmann, É. Fromont, G. Garriga, P. Miettinen, N. Tatti, and V. Tresp, Eds. Cham: Springer International Publishing, 2016, pp. 36–40.

[32] X.-Z. Wang, H.-Q. Zhao, and S. Wang, "The study of unstable cut-point decision tree generation based-on the partition impurity," in *2009 International Conference on Machine Learning and Cybernetics*, vol. 4, 2009, pp. 1891–1897.

[33] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," Jan. 2020, More details here https://www.stratosphereips.org /datasets-iot23. [Online]. Available: https://doi.org/10.5281/zenodo.4743746

[34] H. Chunduri, T. Gireesh Kumar, and P. V. S. Charan, "A multi class classification for detection of iot botnet malware," in *Computing Science, Communication and Security*, N. Chaubey, S. Parikh, and K. Amin, Eds. Cham: Springer International Publishing, 2021, pp. 17–29.

[35] M. Hegde, G. Kepnang, M. Al Mazroei, J. S. Chavis, and L. Watkins, "Identification of botnet activity in iot network traffic using machine learning," in *2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, 2020, pp. 21–27.

[36] M. Lefoane, "Feature selection methods for machine learning pre-processing stage," https://github.com/mlefoane/FeatureSelection/, Apr. 2022, accessed: 2022-04-25.