The Performance of IoT Malware Detection Technique Using Feature Selection and Feature Reduction in Fog Layer

Ban Mohammed Khammas
Department of Computer Networks Engineering, Collage of Information Engineering, AL-Nahrain University, Baghdad, Iraq
e-mail: bankhammas@coie-nahrain.edu.iq

Abstract:

The rapid increase in the number of devices connected to internet (IoT) lead to expansion in the attacks that targeting these devices. One of these dangers attacks is malware which embedded with IoT devices that makes the detection for such malware is extremely challenging. The machine learning is one of the most effective techniques that used to detect different types of attacks in IoT environment. This technique includes three main stages: feature extraction, feature selection, and classification. The feature selection is the most important stage in ML technique because it contributes to minimizing the size of features which significantly accelerate the detection system. In this stage, most researchers trend to use one of three methods; feature selection, feature reduction and hybrid between feature selection and reduction. The present research aims to present a comparative study between the effect of using feature selection method and feature reduction method on the performance of the IoT malware detection system. The results showed that the proposed technique could achieved an accuracy about 97% when using feature selection method only. These results emphasize that feature selection method is more efficient than the feature reduction method in detection IoT malware.

Keywords:
IoT security, IoT malware, Machine Learning, PCA.

Introduction:
Lately, the malware attacks were significantly increase which target the IoT devices more than traditional computers due to the low security properties or even absence of any security system. Kaspersky lab report mentioned that the malware samples which appeared in the first half of 2018 was three times the malware samples that appeared in 2017 [1]. According to AV-TEST’s survey in 2020, more than 350,000 malware samples and potentially unwanted applications (PUA) are register every day [2]. Therefore, there is increasingly needed of a new technique that is able to detect and prevent the spread of malware.

The current study aims to investigate the effect of combining between both feature reduction method and feature selection method in detection IoT malware. This new approach uses a decision tree classifier with Maximum Frequent Patterns (MFPs) of features to select the most important features. In addition, the present work used principal component analysis (PCA) as a feature reduction method and the Gain Ratio as a feature selection method.

Since the defiance technique for IoT should be network based rather than host based [3]. According to [4], the fog node can be used to detect suspicious behaviors in IoT devices. Furthermore, [5] they proposed a machine learning system that could be implemented in fog layer. Therefore, the current proposed technique should be implemented in fog-node.

The rest of this paper is organized as follows: section 2 introduces the related work. Section 3 presents the proposed approach with the dataset collection. Section 4 reports the experimental results with discussion for the main findings. Finally, Section 5 concludes the finding of this work.

**Related Work:**

In recent decade, many researchers try to detect IoT malware using machine learning techniques (ML). Kumar and Lim [3] proposed EDIMA (Early Detection of IoT Malware Network Activity) to detect IoT malware by the scanning packet traffic with the use of ML technique. Many static analysis approach were proposed to detect IoT malware such as, opcode based, control flow graph, n-gram, and string with ML [6]. Darabian et al. [7] used a maximal frequent patterns (MFP) which depended on the opcode sequences features to detect IoT malware. They utilized various types of classifiers namely multilayer perceptron (MLP), K nearest neighbors (KNN), AdaBoost, random forest, support vector machines (SVM), and decision tree. Regarding to dataset, they used 269 IoT goodware and 247 IoT malware. The results achieved an accuracy of 99%. Alhanahnah et al. [8] suggested the unsupervised machine learning technique using a multistage clustering mechanism. They extracted features using n-gram text analysis from a dataset of 5,150 IoT malware samples. This method attained a detection rate around 95.5%.

HaddadPajouh et al. [9] proposed a new approach that used a Recurrent Neural Network (RNN) with an opcode based analysis to detect IoT malware. They utilized three different Long Short Term Memory (LSTM) configurations beside to Information Gain (IG) as a feature selection. The dataset contained 270 goodware and 281 malware. Their findings illustrated that the highest accuracy around 98.18% could be achieved using 2-layer neurons.
Many other researchers adopted Control Flow Graph (CFG) technique. Alasmary et al. [10] used CFG to analyze the IoT malware and android malware using different graph-theoretic features such as; graph size, degree centrality, distribution of shortest path, diameter, radius, .. etc, to find graph-related features. This approach enabled to distinguish between android malware and IoT malware. Alasmary et al. [11] also used CFGs to extract features that can be used to differentiated between IoT malware and Android malware with deep learning-based detection model. The results showed a detection rate around 99.656% with a dataset of 6000 malware samples.

Dynamic analysis is another approach which widely adopted in many research to detect IoT malware. Huda et al. [12] proposed a hybrid approach using multi filter-wrapper framework with different feature selection. This approach exhibited around 99.499% of detection rate. Kargaard et al. [13] used dynamic analysis within a sandbox environment to collect malware samples. This method comprises a conversion of malware binaries to malware images which classified by using a specific method of unsupervised deep neural networks called "Generative Adversarial Network". At the same time, they used 369 malware to be analyzed which captured by honeypots running Dionaea. Phu et al. [14] proposed a new framework to classify IoT malware using MIPS-based system behavior (system call—syscall) obtained from F-Sandbox and machine learning techniques. Besides, they used Chi-square as a feature selection for n-gram features. At the same time, they utilized different machine learning techniques such as Random Forest (RF), Support Vector Machine (SVM), and Naive Bayes (NB) for classification. The results showed that F1-Weight achieved about 97.44%. Ficco [15] proposed a Markov chains model to detect IoT malware. The data set which used in this model contains 24K Malware and 22K benign applications. His results showed that the Markov chains superior over the API call frequency with F-measure around 89%. PA PA et al. [16] used a sandbox system called IoTPOT which represent an IoT honeypot that capture the IoT attacks. moreover, they used over eight CPU architectures for this purpose.

Some researchers used a hybrid technique for feature extraction, Naeem et al. [6] proposed static visual analysis to detect malware using malware image classification system (MICS). They extract hybrid local and global texture features. This technique achieved 97.4% classification rate for 8000 malware.

Based on the previous studies, it can be seen that most research used either feature selection method or feature reduction method for IoT malware detection. The present study aims to explore the effect of combination between two methods on the detection accuracy by comparison with standalone feature selection method.

**Proposed IoT Malware Detection System:**

The fog provides local IoT data processing and storage at IoT devices, rather than sending them to the cloud. The fog offers quicker response and higher quality services, as compared to the cloud. Therefore, fog computing need to be in high level of security in order to provide secure services for many IoT users. Since fog node can be used to detect suspicious behaviors in IoT devices [4, 17]. Based on this keen observation, the proposed approach represented by ML for detecting IoT malware impose to puts near fog nodes. The current study
also analyzes the effect of using feature reduction besides feature selection on the performance of IoT malware detection system compared with the detection system that using feature selection only.

Figure 1: The framework of the proposed IoT malware Detection system using ML.

The procedure of the proposed approach is described in figure 1. the first stage includes collecting dataset which consist of goodware and malware samples of type Executable and Linkable Format (ELF). The features are extracted from these data within feature extraction process where the binary files are transferred to hexadecimal code to be readable. According to many previous researches [18-23], n-gram is considered as one of the most effective features type which used in malware detection. So, the present study adopt the n-gram features to be used instead of the original binary code. The size of each feature is 32-bit, then the Maximum Frequent Patterns (MFPs) of features are used to select the most important features [7].

The feature extraction stage is followed by feature selection stage where the Gain Ratio technique is used as a filter approach. This technique showed a high performance in selecting the most relevant features [24, 25].

In case of using combination of feature selection and feature reduction stage, the Principal Component Analysis (PCA) is used in the proposed approach as feature reduction method. PCA transfers a set of inputs to a linearly uncorrelated set of features using an orthogonal transformation [26]. It decreases the correlated information which produced by the overlapping input instances. The Singular Value Decomposition (SVD) is used in PCA technique to sort the input data in descending order of uncorrelation and importance [27].

The last stage of the proposed method is classification process. The Decision tree (J48) is utilized as a classifier due to its effectiveness in classifying different types of attacks [7, 28-31].
Dataset:

The collection of IoT executable samples that used during the experimental evaluation of the proposed method contain 1000 IoT malware and 1000 goodware samples. All the files of malware and goodware of type Executable and Linkable Format (ELF) binaries. The IoT malware samples were collected from VirusTotal and CyberIOCs (https://freeiocscyberioccs.pro/) [11] which includes the most recent IoT malware. For validating that all samples are a malicious code, they have been uploaded to VirusTotal (http://www.virustotal.com) Threat Intelligence platform. At the same time, the goodware samples were collected from (https://pkgs.org/) [7] and (https://github.com/azmoodeh/) [32]. The goodware samples are ARM-based which utilized in a variety of official IoT App stores such as Pi Store.

Experimental Results:

The experimental procedure was implemented in two operating system(OS) namely, Linux 4.1. and Window 10 that installed in Core i7 CPU, 8 core, 16GB RAM. Furthermore, two Oracle VM VirtualBox version 4.2.16 have been used in current work. Two Oracle VM VirtualBox version 4.2.16 have been used in current work. These two VM’s are used to collect and reprocess the malware samples. First VM running with Windows 10 and the second VM running with CentOS Linux platform. Besides, other programs are also used to prepare the experiments such as; MATLAB 2019b and WEKA 3.9.4 [33] (the machine learning tool).

In order to evaluate the performance of the proposed approach, first the dataset is divided into training and testing groups. The training dataset has been divided in 50% malware and 50% goodware to avoid the imbalance. The training dataset contain 1000 samples (50% goodware and 50% malware). The same partitioned is performed in the testing dataset which also contain 1000 samples (50% goodware and 50% malware).

After preparing the dataset, a measurement of the classification model is performed using confusion matrix. The performance metrics includes: False Negative (FN) (malware samples that are incorrectly predicted as goodware), True Positive (TP) (malware samples that are correctly predicted as malware), False Positive (FP) (goodware samples that are incorrectly predicted as a malware), True Negative (TN) (goodware samples that are correctly predicted as goodware). Based on the aforementioned criteria, the accuracy, f-measure, recall, and precision metric are calculated according to the following equations:

\[
Accuracy = \frac{TP + TN}{TP + FP + TN + FN}
\]

\[
Recall = \frac{TP}{TP + FN}
\]

\[
Precision = \frac{TP}{TP + FP}
\]
\[ F - Measure = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]

The classification IoT malware are shown below in the following figures. Figures 2 & 3 demonstrate the PCA of some samples of the training and testing dataset respectively.

![Figure 2: PCA of some samples of the training dataset.](image1)

![Figure 3: PCA of some samples of the test dataset.](image2)

The classifier model has been build and tested for different number of instances per leaf used in J48. The performances of the J48 are tested in the presence of PCA and without PCA as shown in Figures 4, 5, 6, & 7 respectively. The results showed that FP and FN values for the classifier with PCA are higher than that without PCA. At the same time, it can be seen that TP and TN values for the classifier without PCA are higher than that with PCA.
Furthermore, Figures 8, 9, 10, 11 & 12 demonstrate the precision, f-measure, recall, ROC and accuracy of the testing dataset. The results observed a superiority of precision, f-measure, recall, ROC and accuracy of the classification model which built in the absence of PCA.
Figure 8: The precision value when PCA was used and without using PCA.

Figure 9: The F-Measure value when PCA was used and without using PCA.

Figure 10: The recall value when PCA was used and without using PCA.

Figure 11: The ROC when PCA was used and without using PCA.
Figure 12: The accuracy value when PCA was used and without using PCA.

Finally, Figure 13 demonstrate a comparison between the time needed to train the classifier in presence of PCA and without using PCA. This figure shows that the classifier needs a longer time for training in presence of PCA (45.32 sec.) comparing with system using only feature selection method (0.75 sec.) for 50 instances pear leaf as an example. This attributed to effect of linear transformation of original features onto the space spanned by principal components.
Figure 13: The time measured for the training dataset to build the classifier when PCA was used and without using PCA.

Based on the findings presented in the above Figures, one can easily observe that the PCA is not effective to be used as feature reduction process when combining with feature selection to detect IoT malware in IoT devices. The using of PCA lead to remove some of important features due to the reduction process which contributing in decreasing the performance of the detection system. Besides, the using of feature reduction process consumes more time due to the features transformation into a new feature space by PCA. So, the utilization of feature selection alone showed more effectiveness in detection of IoT malware with an accuracy around (96.7%) especially when the number of instances per leaf is low (2 only).

Table 1 demonstrates the comparison of the proposed method with the existing works. The accuracy results showed by Pajouh et al. [9], and Darabian et al. [7], are higher than the proposed method. However, their techniques need additional time due to the disassemble process which is not suitable to meet the users requirements of IoT network, while the current proposed technique eliminate this additional processing because the features are extracted directly from raw binary file. Besides, their results are not reflecting the real accuracy due to small dataset that they used.

Table 1: Comparison with the existing works.

| Method                  | Types of Features | Feature reduction or feature selection | Dataset (Malware (M) / Goodware (G)) | Results      |
|-------------------------|-------------------|----------------------------------------|--------------------------------------|--------------|
| Pajouh et al. [9]       | Recurrent Neural Network (RNN) | OpCodes                                | 281 M / 270 G                       | Acc. = 98.18% |
| Darabian et al. [7]     | ML                | OpCodes                                | 247 M / 269 G                       | Acc. = 99%   |
| Proposed method         | ML                | Binary n-gram                          | 1000 M / 1000 G                     | Acc. = 96.7% |

Conclusion:

Nowadays, devices connecting to the internet are widely spread in all over the world. This expansion led to more attacks that intendeds these devices. In this paper, we examined the potential of using a combination of feature selection and feature reduction with J48 to detect IoT malware. The experiment shows the superiority of the classifier that used feature selection with Maximum Frequent Patterns only over that used combining selection and reduction to the features. The best results achieved around 96.7% of accuracy using feature selection with
Maximum Frequent Patterns only. Future research will expand the proposed approach to examine the other machine learning techniques for IoT malware analysis.

References:

1. Kaspersky, New trends in the world of IoT threats. 2018.
2. AV-TEST, Malware 2020.
3. Kumar, A. and T.J. Lim, EDIMA: Early Detection of IoT Malware Network Activity Using Machine Learning Techniques. arXiv preprint arXiv:1906.09715, 2019.
4. Diro, A. and N. Chilamkurti, Leveraging LSTM networks for attack detection in fog-to-things communications. IEEE Communications Magazine, 2018. 56(9): p. 124-130.
5. Alrashdi, I., et al. AD-IoT: anomaly detection of IoT cyberattacks in smart city using machine learning. in 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). 2019: IEEE.
6. Naeem, H., B. Guo, and M.R. Naeem. A light-weight malware static visual analysis for IoT infrastructure. in 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD). 2018: IEEE.
7. Darabian, H., et al., An opcode-based technique for polymorphic Internet of Things malware detection. Concurrency and Computation: Practice and Experience, 2019: p. e5173.
8. Alhanahnah, M., et al. Efficient signature generation for classifying cross-architecture iot malware. in 2018 IEEE Conference on Communications and Network Security (CNS). 2018: IEEE.
9. HaddadPajouh, H., et al., A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting. Future Generation Computer Systems, 2018. 85: p. 88-96.
10. Alasmary, H., et al. Graph-based comparison of IoT and android malware. in International Conference on Computational Social Networks. 2018: Springer.
11. Alasmary, H., et al., Analyzing and detecting emerging Internet of Things malware: a graph-based approach. IEEE Internet of Things Journal, 2019. 6(5): p. 8977-8988.
12. Huda, S., et al., A hybrid-multi filter-wrapper framework to identify run-time behaviour for fast malware detection. Future Generation Computer Systems, 2018. 83: p. 193-207.
13. Kargaard, J., et al. Defending IT systems against intelligent malware. in 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT). 2018: IEEE.
14. Phu, T.N., et al., A Novel Framework to Classify Malware in MIPS Architecture-based IoT Devices. Security and Communication Networks, 2019.
15. Ficco, M. Detecting IoT Malware by Markov Chain Behavioral Models. in 2019 IEEE International Conference on Cloud Engineering (IC2E). 2019: IEEE.
16. Pa, Y.M.P., et al., Iotpot: A novel honeypot for revealing current iot threats. Journal of Information Processing, 2016. 24(3): p. 522-533.
17. Wazid, M., et al., IoMT Malware Detection Approaches: Analysis and Research Challenges. IEEE Access, 2019.
18. Ismail, I., et al., *Incorporating known malware signatures to classify new malware variants in network traffic*. International Journal of Network Management, 2015. 25(6): p. 471-489.

19. Khammas, B.M., et al., *Metamorphic malware detection based on support vector machine classification of malware sub-signatures*. TELKOMNIKA (Telecommunication Computing Electronics and Control), 2016. 14(3).

20. Khammas, B.M., I. Ismail, and M. Marsono, *Pre-filters in-transit malware packets detection in the network*. Telkomnika, 2019. 17(4).

21. Khammas, B.M., et al. *Accuracy improved malware detection method using snort sub-signatures and machine learning techniques*. in 2018 10th Computer Science and Electronic Engineering (CEEC). 2018: IEEE.

22. Khammas, B.M., et al., *Feature selection and machine learning classification for malware detection*. Jurnal Teknologi, 2015. 77(1).

23. Zhang, H., et al., *Classification of ransomware families with machine learning based on N-gram of opcodes*. Future Generation Computer Systems, 2019. 90: p. 211-221.

24. Carlin, D., P. O’Kane, and S. Sezer, *Dynamic analysis of malware using run-time opcodes*, in *Data analytics and decision support for cybersecurity*. 2017, Springer. p. 99-125.

25. Pehlivan, U., et al. *The analysis of feature selection methods and classification algorithms in permission based Android malware detection*. in 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS). 2014: IEEE.

26. Zhang, J., et al., *Sensitive system calls based packed malware variants detection using principal component initialized MultiLayers neural networks*. Cybersecurity, 2018. 1(1): p. 10.

27. Mohamad-Saleh, J. and B.S. Hoyle, *Improved neural network performance using principal component analysis on Matlab*. International journal of the computer, the internet and Management, 2008. 16(2): p. 1-8.

28. Moon, D., et al., *DTB-IDS: an intrusion detection system based on decision tree using behavior analysis for preventing APT attacks*. The Journal of supercomputing, 2017. 73(7): p. 2881-2895.

29. Patil, D.R. and J. Patil, *Malicious URLs detection using decision tree classifiers and majority voting technique*. Cybernetics and Information Technologies, 2018. 18(1): p. 11-29.

30. Aljawarneh, S., M.B. Yassein, and M. Aljundi, *An enhanced J48 classification algorithm for the anomaly intrusion detection systems*. Cluster Computing, 2017: p. 1-17.

31. Alhawi, O.M., J. Baldwin, and A. Dehghantanha, *Leveraging machine learning techniques for windows ransomware network traffic detection*, in *Cyber Threat Intelligence*. 2018, Springer. p. 93-106.

32. Azmoodeh, A., A. Dehghantanha, and K.-K.R. Choo, *Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning*. IEEE Transactions on Sustainable Computing, 2018. 4(1): p. 88-95.

33. Weka, *Open Source Machine Learning Software in Java*. 2020.