PAM Clustering Aided Android Malicious Apps Detection

Nibras Talib Mohammed 1a, Mohsin Hasan Hussein2b and Abbas Jabber Rashid3c

1Department of statistics, College of Administration and economy, University of Kerbala, Iraq.
2Department of Computer Science, College of Science, University of Kerbala, Iraq.
3Ministry of Interior, Civil Defense Director at Kerbala, Iraq.

a Corresponding author:nbrass.t@uokerbala.edu.iq.
b mohsin.h@uokerbala.edu.iq
ca82bbas@yahoo.com

Abstract. The exponential growth of android contrivances has attracted cybercriminals strongly and dramatically. The applications existed in the android market represented an attack surface owing to the lack of security mechanisms applied by the Google play store. Additionally, downloading apps from unofficial sources lead to a further security threat. Any mobile application requests several permissions to access users' data to run the app. Thus, attackers exploited this feature in compromising users' sensitive data. This motivated several researchers to investigate security mechanisms to detect Android malware based on this feature utilizing machine learning techniques, particularly classification techniques. However, This research proposes a permission-based android malware detection framework using a clustering algorithm. Further motivation for this research is that large datasets labeling is a tough mission. Therefore, Our work will contribute to android malware detection as well as android apps datasets labeling. PAM (Partitioning Around Medoid) clustering has been exploited for this purpose since its less affected by outliers or other extreme values. The most significant features have been selected as an input to the clustering algorithm to enhance its results. The results depicted that our clustering algorithm was able from grouping our dataset into two categories malevolent and genuine apps. Moreover, our result has been validated by evaluation standard F-Measure, which is counting for 0.86 for 40 attributes subset, while it is 0.88 for 30 features subset. This reveals a good performance level of permission-based android malware detection and android applications datasets labeling into malware and good ware.

Keywords: Android malware, PAM clustering, Permission, Feature selection.

1. Introduction
In the gone decade, mobile ecosystems have gained great popularity. Thus cause a huge incrimination in the number, as well as the variety of the applications, run on those devices. Owing to the trivial progress on mobile phone functionality, it fascinated a wider range of population [1]. The rapid upgrading of mobile phones and apps produced a huge vicissitude in people's way of performing tasks in various life fields, for instance conducting online meetings, online business via their phones[2]. The most popular applications markets today are Google Play and Apple's app store. Those markets have demonstrated a steep growth in terms of both the applications offered to the users as well as the downloads performed by users [3]. There are security vulnerabilities on mobile computing allowing the attackers to target sensitive information when the legitimate person accesses his/her assets [4].

Moreover, The diffusion of malicious apps is caused by market providers' policy[5]. For instance, Apple's app store applies security checks for any application before releasing it on its market [5]. However, Google’s Play Market offers a higher level of freedom in uploading applications to mobile apps developers. Even though, those apps can be ejected by the market providers in case of reporting malevolent activity [6]. Specifically, the android operating system permits the remote ejection of malicious software from the users' devices. The post-detection mechanism leads to the necessity for earlier discovery for any malignant software. Android Malware detection can be conducted even by static or dynamic analysis [7]. In static form, the analysis is conducted by reverse-engineering the app without the need to run it [8]. The analysis process is based on files with .apk extension especially Android Manifest.xml and classes.dex [9]. However, carrying out dynamic analysis requires running the application in a controlled environment to analyze its execution traces[10]. Dynamic analysis requests more computational resources than static analysis including execution time and memory space [11]. Android apps can be run based on the employment of function awarded by the operating system. Therefore, the app should be granted several permissions approved by the device utilizer. Those sanctions should be requested from the user during the app installation. [12] comprises a list of permission that might be requested by an android application.

Permissions represent application capabilities in a device operating system. Thus, it can be exploited by hackers in accessing users’ crucial data by the mean of the application[13]. Since the application doesn’t need to be run to analyze its permissions, this form of analysis of the malevolent software can be considered as static analysis. In the instant study, Permission-based malware detection for the android operating system has been proposed. In a permission-based analysis of malevolent applications, there are two crucial phases including feature selection methods and machine learning algorithms. Even though classification algorithms are capable of presenting satisfactory outcomes, they demanded a labeled data sets. Since labeling huge datasets is a time-consuming mission, clustering techniques are capable of discovering patterns to classify the apps into two clusters.

Thus, we proposed utilizing unsupervised machine learning (PAM clustering) for permission-based android malware detection purposes.

2. Related Work
Several prior research utilized machine learning for android malignant software detection. For instance [14] depends on permission as a data input to their static analysis. It employed four feature cull methods for shrinking dataset size counting for Gain Ratio Attribute, Cfs Subset Consistency Subset Relief, and Attribute Evaluator. Additionally, Bayesian relegation, J48 Decision, Arbitrary Forest, Regression Tree, Sequential Minimal Optimization have been performed as classification algorithms. The fulfillment reveals that Random Forest and J48 decision tree classification algorithms presented the highest performance with all chosen feature extraction approaches. Further research considered permissions as a backbone of its analysis is [15]. Four classification algorithms are utilized including K-NN, Naïve Bayes, SVM, and Decision Tree. In this research best accuracy introduced by the Naïve Bayes algorithm. SigPID was a new detection approach based on electing only the significant permissions, which can have an impact on malware detection. In this study, only 22 permission has been selected as entries to the classification algorithms. The method followed in this research was capable of 93.62% of malware detection in their dataset [16]. In [17] permissions and applications source code were exploited as data entry to its classification. It demonstrated that source code classification introduced better results than permission-based classification. The authors of [18] presented a hybrid feature extraction technique based on Rough Set Quick algorithm with a Community Detection scheme. The contribution of this research dedicated to featuring reduction and exposes higher accuracy than existing feature cull techniques. On the other hand, other research exploited clustering techniques in android malware detection. One of those research was [19], which utilized the Fuzzy C-Means clustering algorithm in detecting malicious applications. The extracted features were network traffic. The results revealed that there is 13% contradiction between the labeled dataset and the experimental findings. The authors of [20] proposed a clustering system based on malevolent payload code in android malware detection. Their results depicted 0.90 and 0.75 of precision and recall respectively. however, our research key focus is on exploiting PAM clustering for permission-based android malware detection.

3. **Methodology**

The methodology of this research comprises of three phases. In the first phase, significant features are selected utilizing the information gain algorithm to enhance the clustering results. The second phase is the PAM clustering algorithm implementation, which is the improved version of the widely used K-mean algorithm. It is applied to cluster the applications in the dataset into malicious and benign. The evaluation process is the last step of our work. The overview design of the proposed model is presented in Figure. (1).

![FIGURE 1. Android malware detection framework](chart.png)
i. Dataset

The dataset used in this research is from [21]. It consists of 398 applications with 199 malicious apps and 199 non-malware apps. The files have been extracted from .apk files of apps. Then permissions have been extracted from the manifest.xml file for each app. Each app was represented by a binary vector for all the permissions which might be requested by any app. The value (1) is used if the app uses this permission, while (0) represented unused permission. Each app was stamped by (1) if it is malignant apps and (0) for benign apps. This label has been excluded from unsupervised machine learning. Then, it has been returned for validation purposes.

ii. Feature Selection

The efficiency of machine learning depends on the appropriate feature selection method since it interacts with a used algorithm and its results. If a dataset included a huge number of features (permissions) and some of them are redundant. This might cause several issues, for instance, raising run time or model complexity, misleading the algorithm, and etc [22]. Those problems can be much complex when conducting machine learning on mobile devices due to restricted capabilities of those devices in terms of processing, storage media, and battery power. For this reason, appropriate feature selection is crucial for efficient and faster malware detector. Statistical computing is applied for our data after that Information Gain algorithm has been applied to select the most effective features for our malware detector.

Statistical Computing

The used dataset included (331) permissions and some of them were not utilized by any of the apps in the dataset or some of them used by few applications. Thus, we performed statistical computing to shrink features by deleting features with all zeros or with at most three 1’s, since it might have no impact on the clustering algorithm.

Information Gain method

This feature selection method depends on the amount of information that can be gained based on the entropy of the feature. The best feature is the one that gives the highest value of the gain [23]. After deleting features based on Statistical computing, the information Gain method has been used for ranking the rest of features. This method selects the k best attributes from the features of the android application manifest.xml file. A gain of any feature (x) in the dataset (Y) can be calculated by the following formula from [22]:

\[ \text{Gain}(x, Y) = \sum_{y \in Y} \frac{P(y) \log_{2} \frac{P(y)}{P(x,y)}}{P(x)} \]
Gain (Y, x) = Entropy (Y) - $\sum |YV| \text{ Entropy}(YV) , V \in \text{Values(x)} |Y|$ \hspace{1cm} (1)

iii. **PAM Clustering (Partitioning Around Medoid)**

K-medoids is a basic method for clustering in PAM, which is related to the k-means and medoid shift algorithms. However, the k-medoids method is more robust than k-means in the presence of noise and outliers, because a medoid is less influenced by outliers or other extreme values than a mean [24]. In the PAM algorithm, the K objects are selected as medoids (centroids) of clusters, remaining objects of data are assigned to the closest cluster, where “closest” means closest to the medoid of the cluster [25].

The last step of PAM clustering is swapping the current medoids with new random non-medoid objects provided that new cost is better than the current cost and repeating the step until it meets the stop condition. The algorithm steps are illustrated below:

**PAM Clustering Algorithm**

**Input:**
- Data set D with n data points (objects)
- Number of clusters K

**Output:**
K clusters

**Begin**
- Select k initial medoids from D.
- Assign each data object to the closest medoid.
- Calculate the cost1 (sum of the distance between medoids and its’ cluster objects).

Repeat while the cost decreases
- Randomly select Non-medoid object as new medoid.

For each medoid
- Swap current medoid with new medoid.
- Assign each object to the closest medoid.
- Recalculate the cost (cost2).
- If cost1 < cost2 Then Undo the swap

Else cost1 = cost2 and exit For.

**End For**

**End while**

**End**

iv. **Evaluation**
There are several evaluation standards used in evaluating clustering techniques' results. One of these standards is F-measure [26], which will be utilized by this research. F-measure can be computed based on precision and recall values. Precision reflects how many instances are predicted in the correct cluster concerning the cluster size. Recall indicates the fraction of the correctly clustered instances for the whole instances of True positive and false negative. Precision and Recall can be calculated as follows:

\[
\text{Precision} = \frac{TP}{TP + FP} \quad \text{(2)}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \quad \text{(3)}
\]

Where (TP) represents truly identified benign applications as non-malware. Applications that have false identification as a good ware while they are malware in reality, are (FP). The truly identified applications as malware are (TN), while the benign applications, which wrongly determined as malicious are (FN). F-Measure is given as:

\[
\text{F-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{(4)}
\]

4. Results and Discussion

As mentioned before that our dataset consists of 400 applications, 199 malware apps, and 199 good ware apps.

In this research statistical computing was performed firstly to decrease the dataset features from 331 to 68 features. Secondly, the Information Gain algorithm has been used to eject redundant or unnecessary features. This algorithm results assist us in selecting only the most significant 30 and 40 features' subsets. Then PAM clustering was implemented on both subsets with the number of medoids (k) is 2, to produce the results as shown in table 1:

|                | Size of Cluster 1 | Size of Cluster 2 | Outliers |
|----------------|-------------------|-------------------|----------|
| Selected 40 features | 197               | 190               | 11       |
| Selected 30 features  | 206               | 177               | 15       |

This research evaluation process built on the comparison between our clustering algorithm results and the labeled dataset. Table (2) demonstrates the validation results. For the 40 attributes subset, the clustering algorithm truly identified 173 malware applications as malware (TN) in cluster 1. However, 24 applications have false identification as a good ware while they are labeled as malware in the dataset (FP). For the real benign apps in the data set, 164 of them were truly identified as non-malware(TP) and 26 of them have wrongly determined malicious (FN) in cluster 2.
PAM clustering results with 30 features subset identified 184 apps as (TN), and 22 apps determined (FP) in cluster 1. While 165 apps identified as (TP) and 12 apps determined as (FN) in cluster 2. As a result, for 40 features subset the True Positive Rate (TPR) is 0.91 and False Positive Rate (FPR) is 0.1, while for 30 features subset, TPR is 0.93 and FPR is 0.1.

| TABLE 2: Validation Results |
|-----------------------------|
| Size of Cluster 1 | Size of Cluster 2 | Outliers |
|-------------------|-------------------|----------|
| Selected 40 features | Malware 173 | non-malware 24 | Malware 26 | non-malware 164 | 11 |
| Total = 197 | Total = 190 |
| Selected 30 features | Malware 184 | non-malware 22 | Malware 12 | non-malware 165 | 15 |
| Total = 206 | Total = 177 |

Other evaluation measures including precision, recall are 0.87 and 0.86 respectively for 40 features subset. For 30 features subset precision is 0.83 and recall is 0.93. F-Measure is 0.86 for 40 attributes subset, while it is 0.88 for 30 features subset. It can be observed that F-Measure values for both subsets are convergent. This indicates that shrinking the permissions from 40 to 30 has little impact on the clustering algorithm results. Furthermore, these results indicate a good performance of android malware detection based on permission and labeling android applications' datasets into malicious and benign.

5. Conclusions and future work

In this research, unsupervised machine learning has been exploited to cluster android applications to determine whether they are malevolent or legitimate. Permissions are the features, which have been used as input to the clustering algorithm. In the first stage, the Information Gain algorithm has been applied to shrink the attributes dimension in the dataset. Then, we utilized the resulted subsets as an input to the PAM clustering algorithm. To validate the proposed model, TPR and FPR, precision, recall, and F-Measure have been computed. Consequently, the results indicated an acceptable level of clustering of any android applications dataset to malware and benign apps. Regarding future work, other clustering algorithms will be utilized for android malware detection and android apps dataset labeling.
References

[1] M. S. Rana, C. Gudla, and A. H. Sung, Evaluating Machine Learning Models on the Ethereum Blockchain for Android Malware Detection, vol. 998, no. July. Springer International Publishing, 2019.

[2] Z. Yuan, Y. Lu, and Y. Xue, “Droiddetector: Android malware characterization and detection using deep learning,” Tsinghua Sci. Technol., vol. 21, no. 1, pp. 114–123, 2016, doi: 10.1109/TST.2016.7399288.

[3] U. Pehlivan, N. Baltaci, C. Acarturk, and N. Baykal, “The analysis of feature selection methods and classification algorithms in permission based Android malware detection,” IEEE SSCI 2014 2014 IEEE Symp. Ser. Comput. Intell. - CICS 2014 2014 IEEE Symp. Comput. Intell. Cyber Secur. Proc., no. December, 2014, doi: 10.1109/CICYBS.2014.7013371.

[4] S. Y. Yerima, S. Sezer, and I. Muttik, “High accuracy android malware detection using ensemble learning,” IET Inf. Secur., vol. 9, no. 6, pp. 313–320, 2015, doi: 10.1049/iet-ifs.2014.0099.

[5] G. Suarez-Tangil, S. K. Dash, M. Ahmadi, J. Kinder, G. Giacinto, and L. Cavallaro, “DroidSieve: Fast and accurate classification of obfuscated android malware,” CODASPY 2017 - Proc. 7th ACM Conf. Data Appl. Secur. Priv., pp. 309–320, 2017, doi: 10.1145/3029806.3029825.

[6] M. Ghafari, P. Gadient, and O. Nierstrasz, “Security Smells in Android,” Proc. - 2017 IEEE 17th Int. Work. Conf. Source Code Anal. Manip. SCAM 2017, vol. 2017-Octob, pp. 121–130, 2017, doi: 10.1109/SCAM.2017.24.

[7] L. Onwuzurike, M. Almeida, E. Mariconti, J. Blackburn, G. Stringhini, and E. De Cristofaro, “A Family of Droids-Android Malware Detection via Behavioral Modeling: Static vs Dynamic Analysis,” 2018 16th Annu. Conf. Privacy, Secur. Trust. PST 2018, no. Pst, 2018, doi: 10.1109/PST.2018.8514191.

[8] A. Bacci, A. Bartoli, F. Martinelli, E. Medvet, F. Mercaldo, and C. A. Visaggio, “Impact of Code Obfuscation on Android Malware Detection based on Static and Dynamic Analysis,” ICISSP 2018 - Proc. 4th Int. Conf. Inf. Syst. Secur. Priv., vol. 2018-Janua, no. Icissp, pp. 379–385, 2018, doi: 10.5220/0006642503790385.

[9] L. Onwuzurike, M. Almeida, E. Mariconti, J. Blackburn, G. Stringhini, and E. De Cristofaro, “A Family of Droids-Android Malware Detection via Behavioral Modeling: Static vs Dynamic Analysis,” 2018 16th Annu. Conf. Privacy, Secur. Trust. PST 2018, no. ii, 2018, doi: 10.1109/PST.2018.8514191.

[10] M. Y. Wong and D. Lie, “IntelliDroid: A Targeted Input Generator for the Dynamic Analysis of Android Malware,” no. February, pp. 21–24, 2017, doi: 10.14722/ndss.2016.23118. [11] A. Damodaran, F. Di Troia, C. A. Visaggio, T. H. Austin, and M. Stamp, “A comparison of static, dynamic, and hybrid analysis for malware detection,” J. Comput. Virol. Hacking Tech., vol. 13, no. 1, 2017, doi: 10.1007/s11416-015-0261-z.

[12] F. Tchakounté and P. Dayang, “System Calls Analysis of Malwares on Android,” Int. J. Sci. …, vol. 2, no. 9, pp. 669–674, 2013.

[13] L. Taheri, A. F. A. Kadir, and A. H. Lashkari, “Extensible android malware detection and family classification using network-flows and API-calls,” Proc. - Int. Carnahan Conf. Secur. Technol., vol. 2019-Octob, no. October, 2019, doi: 10.1109/CCST.2019.8888430.

[14] S. Dighe, A. Jondhale, and D. J. Salunke, “Permission Based Android Malware Detection,” vol. 4, no. 6, pp. 33–36, 2016.

[15] T. S. Barhoom and M. I. Nasman, “Malware Detection Based on Permissions on Android Plattform Using Data Mining,” J. Eng. Res. Technol., vol. 3, no. 3, pp. 51–57, 2016.

[16] J. Li and ; Lichao Sun ; Giben Yan ; Zhiqiang Li ; Witawas Srisa-an ; Heng, “Significant Permission Identification for Machine-Learning-Based Android Malware Detection,” IEEE Trans. Ind. Informatics, vol. 14, no. 7, 2018.

[17] K. R. Machine and E. Engineering, “This is a repository copy of Machine learning aided Android malware classification . White Rose Research Online URL for this paper: Version : Accepted Version Article : Article available under the terms of the CC-BY-NC-ND licence Machine learning aided m,” 2017.

[18] A. Bhattacharya and R. T. Goswami, “A Hybrid Community Based Rough Set Feature Selection Technique in Android Malware Detection,” Lect. Notes Networks Syst., vol. 18, pp. 249–258, 2018, doi: 10.1007/978-981-10-6916-1_23.

[19] A. Feizollah, N. B. Anuar, and R. Salleh, “Evaluation of Network Traffic Analysis Using
Fuzzy C-Means Clustering Algorithm in Mobile Malware Detection,” Adv. Sci. Lett., vol. 24, no. 2, pp. 929–932, 2018, doi: 10.1166/asl.2018.10660.

[20] Y. Li, J. Jang, X. Hu, and X. Ou, “Android Malware Clustering Through Malicious Payload Mining,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 10453 LNCS, pp. 192–214, 2017, doi: 10.1007/978-3-319-66332-6_9.

[21] C. Urcuqui and A. Navarro, "Machine learning classifiers for android malware analysis”. In Communications and Computing (COLCOM), 2016 IEEE Colombian Conference on (pp. 1-6). (2016, April). IEEE.

[22] Z. Aung and W. Zaw, “Permission-Based Android Malware Detection,” Int. J. Sci. Technol. Res., vol. 2, no. 3, pp. 228–234, 2013.

[23] J. Novaković, P. Srbc, and D. Bulatović, “Toward optimal feature selection using ranking methods and classification algorithms,” Yugosl. J. Oper. Res., vol. 21, no. 1, pp. 119–135, 2011, doi: 10.2298/YJOR1101119N.

[24] H. Jiawei and K. Micheline,“Data Mining: Concepts and Techniques”. Elsevier. 2006.

[25] M. Dunham , "Data Mining, Introductory and Advanced Topics", Prentice Hall.2002.

[26] A. A. A. Samra, K. Yim, and O. A. Ghanem, “Analysis of clustering technique in android malware detection,” Proc. - 7th Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput. IMIS 2013, pp. 729–733, 2013, doi: 10.1109/IMIS.2013.111.