Journal of Information Technology Management

Print ISSN: 2008-5893

Online ISSN: 2423-5059

Android Malware Category and Family Identification Using Parallel Machine Learning

Ahmed Hashem El Fiky*

*Corresponding Author, M.Sc. in Systems and Computers Engineering, Department of Systems and Computers Engineering, Faculty of Engineering Al-Azhar University, Cairo, Egypt. E-mail: 0x4186@gmail.com

Mohamed Ashraf Madkour

Professor, Department of Systems and Computers Engineering, Faculty of Engineering Al-Azhar University, Cairo, Egypt. E-mail: mamadkour@azhar.edu.eg

Ayman El Shenawy

Assistant Professor, Department of Systems and Computers Engineering, Faculty of Engineering Al-Azhar University, Cairo, Egypt; Software Engineering and Information Technology, Faculty of Engineering and technology, Egyptian Chinese University, Cairo, Egypt. E-mail: eaymanelshenawy@azhar.edu.eg

Abstract

Android malware is one of the most dangerous threats on the Internet. It has been on the rise for several years. As a result, it has impacted many applications such as healthcare, banking, transportation, government, e-commerce, etc. One of the most growing attacks is on Android systems due to its use in many devices worldwide. De-spite significant efforts in detecting and classifying Android malware, there is still a long way to improve the detection process and the classification performance. There is a necessity to provide a basic understanding of the behavior displayed by the most common Android malware categories and families. Hence, understand the distinct ob-jective of malware after identifying their family and category. This paper proposes an effective systematic and functional parallel machine-learning model for the dynamic detection of Android malware categories and families. Standard machine learning classifiers are implemented to analyze a massive malware dataset with 14 major mal-ware categories and 180 prominent malware families of the CCCS-CIC-AndMal2020 on dynamic layers to detect Android malware categories and families. The paper ex-periments with many machine learning algorithms and compares the proposed model with the most recent related work. The results indicate more than 96 % accuracy for Android Malware Category detection and more than 99% for Android Malware family

detection overperforming the current related methods. The proposed model offers a highly accurate method for dynamic analysis of Android malware that cuts down the time required to analyze smartphone malware.

Keywords: Android Malware, Malware Analysis, Malware Category Classification, Malware Family Classification, Malware Dynamic Analysis.

Journal of Information Technology Management, 2022, Vol. 14, No.4, pp. 19-39
Published by University of Tehran, Faculty of Management
doi: https://doi.org/ 10.22059/jitm.2022.88133
Article Type: Research Paper

Received: April 13, 2022
Received in revised form: May 25, 2022
Accepted: June 28, 2022
Published online: July 13, 2022

Introduction

As it is well known, Android is an open-source system for various devices, including mobile phones, tablets, TVs, cars, etc. due to it being open-source, many companies adopted it in their products. The android architecture consists of four layers: (i) Linux kernel: the most significant one representing the Android system's core. It maintains the hardware operations such as memory, power, drivers, the network stack, security settings, shared libraries, and hardware abstraction, as well as the operating system services it delivers. In addition, to providing native libraries that control data processing. (ii) Libraries: it is the native layer that offers open-source libraries, such as the surface manager and media framework. (iii) Application framework includes the android APIs (classes and interfaces used to create Android applications). This layer interacts with the device's running applications and handles the device's fundamental functionalities. The most significant applications in this layer are the activity manager, content provider, telephone manager, location manager, and resources manager. (iv) Application Layer: This is the highest layer where users can access the phone's functionality. It allows users to make calls, manage contacts, send messages, and browse the web. Key programs are delivered such as email, calendar, browser, maps, contacts, SMS, gallery, etc.

Recently, Android became the dominant operating system in the smartphone markets, with market shares at all levels. It grew from a minor player when it debuted in 2010 to power 87 percent of smartphones globally in 2019. It is expected to increase to 87.4 percent in 2023 (Statista, et al. 2022). Furthermore, it controls more than 41.42 percent of the global operating system market (StatCounter Global Stats, et al. 2022). In parallel, the Android malware industry is increasing, with nearly 12,000 new cases of Android malware being reported every day (Lueg, C. et al. 2022). It uses adaptive and new technology to improve its attack effectiveness, which calls for the urgent need for unquestionably better malware detection tools. Users' data and phone hardware are protected by Android security. Android security relies on the Linux kernel, application sandboxing, app signing, and application-defined and

user-granted permissions. The Linux kernel is widely trusted in high-security contexts. Due to its open nature, Linux security is continually improved by security experts, developers who address security problems, and attackers who discover new vulnerabilities. It also allows developers to delete insecure and unwanted kernel sections. By default, programs run in a sandbox, separating their processes and data. Android assigns a unique Linux user ID to each program upon installation. The developer's certificate must sign the .apk file to use the program signing functionality. Apps signed by the same certificate may share a UID. It also enables the system to give or refuse signature-level rights; the system grants signature-level permissions to an app whose certificate matches another app's permission declaration. Finally, Android's permission model protects the phone's resources and functionalities by limiting access to applications that have been given proper capabilities. No app by default has access to the phone's hardware, software, functionalities, or data. App developers must define permissions necessary for app functioning. Android has around 130 built-in permissions and enables app developers to add additional ones through the dynamic permissions option. The built-in permissions are divided into four categories: regular, dangerous, signature, and system.

Android malware is malicious software designed to disrupt Android-based smartphones' functionality by undertaking illegal acts. The most frequent malware categories are adware (Ideses, I., et al. 2014), backdoor, file infector, potentially unwanted application (PUA), ransomware (Ko, J., et al. 2019), riskware, scareware (Mylonas, A., et al. 2012) , and Trojan (Faghihi, F., et al. 2018). Each malware category contains characteristics that separate it from the others and have its own malware families. Each Android Malware belongs to a specific category and its related family. Identifying the android malware category and family helps cybersecurity researchers and Anti-Malware companies to take fast, suitable actions. Hence it decreases the harm resulting from this malware. Some of the actions could be taken for protection, such as adding rules in mobile anti-malware solutions to block the malicious URLs of the Android Malware category from adware and adding rules to block unknown network protocols or block specific ports. As a result, detecting and mitigating malware samples as soon as they are found is the only approach to eliminate this threat. The key to achieving this is having a basic understanding of Android malware categories and families. To better understand the type of features used for detection, classified into static and dynamic features. Features that may be identified or used without the program being executed are known as "static features". Static features include the package name, the application size, access rights, and a list of APIs. On the other hand, dynamic features need the application to be executed, for instance, network traffic, SMS send/receive, resource usage, system logs, and I/O activities. On the other side, the analysis methods could be static, dynamic, or hybrid. Furthermore, the used techniques could be classified into two classes: model-based and analysis-based. The model-based approach may use similarity-based algorithms and evasion-

based methods in addition to machine learning algorithms. At the same time, the analysis-based techniques may utilize statistical, signature, or visualization approaches.

Machine learning approaches have been widely used to classify and identify applications, focusing on malware identification. Machine learning in Android Malware detection reduces the time and effort required to manually create and update detection patterns (Arp, D., et al. 2018). This paper applies machine-learning techniques to discover and identify the Android Malware category and family. Furthermore, installing third-party applications is the primary source of security breaches in the Android platform. More complex static and dynamic analysis approaches have lately been presented, seeking to combine the advantages of traditional methods while improving on their shortcomings.

After careful study of the previous research related to detecting Android malware, a set of limitations were identified as follows: 1) Most researchers concentrate only on the static analysis of Android Malware, where it can bypass the detection of static analysis by obfuscation techniques; 2) Most studies focus only on detecting Android malware apps as benign or malware and ignore the importance of the malware category and/or malware family, which decreases the chance to perform the proper action to protect Android devices; 3) A comprehensive comparison between different approaches is missing.

Therefore, the main contribution of this paper can be summarized as follows:

- An effective systematic and functional parallel machine-learning model is proposed for detecting and classifying Android malware categories and families dynamically to take advantage of the close relationship.
- The performance of the proposed model is assessed using various standard machine-learning classifiers such as Decision tree (J48), Naive Bayesian (NB), Support Vector Machine (SVM), AdaBoost (AB), Logistic Regression (LR), K-Nearest Neighbor (KNN), Random Forest (RF), and Multilayer Perceptron (MLP).
- Finally, the proposed model performance is compared with other models trying to solve the same problem.

The remainder of the paper is structured as follows. Section 2 summarizes the related work of this research point. Section 3 presents our proposed methodology. Section 4 interprets experimental results. Section 5 provides a discussion and limitations of our proposed model. Section 6 concludes with recommendations for future work.

Literature Review

This section presents some of the related work to our proposed approach. For instance, in (El Fiky, A. H., et al. 2021), the authors survey recent android malware detection techniques for android devices. They listed the pros and cons of each approach and stated the details of the

user database, like the number of benign and malware applications and the source of the dataset. Also, in (El Fiky, A. H., et al. 2021), the authors also apply machine-learning algorithms (RF, J48, NB) to detect smartphone malware applications based on the static malware analysis technique. They detect Android malware apps with an accuracy that reached 98.4%. The authors of (Ma, Z., et al. 2019) built three distinct types of datasets based on Machine Learning using a Control Flow Graph (CFG) and API information taken from Android Malware applications. To achieve 98.98 % detection precision, DNN, LSTM, and C4.5 algorithms were used to run tests on 10,010 benign and 10,683 malicious applications. Another approach is presented in (Arslan, R. S., et al. 2019), where the authors employed static and code analysis approaches to identify the spare permissions looked by some apps to do suspicious actions. They were able to detect Android Malicious apps with an accuracy of 91.95 %. Moreover, to assess several machine learning classifiers (NB, J48, SVM, RF, SL), (Yerima, S. Y., et al. 2019) employed static information such as permissions, intents, API calls, and date of appearance derived from date-labeled benign and malware datasets. This classification is followed by GefDroid (Fan, M., et al. 2019), which uses unsupervised learning to accomplish graph embedding examining Android malware families. The program semantics were abstracted using a fine-grained behavioral model to build a set of sub-graphs. Also, AndroDialysis (Feizollah, A., et al. 2017) uses intentions and permissions to identify the malware. AndroDialysis compared the effectiveness of employing permissions with intent and found that the Bayesian Network approach produces a higher detection rate and shorter detection time.

In (Lou, S., et al. 2019), TFDroid utilized SVM to detect malware in android apps by looking at the source, sink, and description. Clustering applications identify outliers into distinct domains depending on the report. Cross-validation is performed on a small dataset, and only benign applications are utilized for training the classifier, resulting in a 93.65% accuracy in recognizing the malicious program. To detect unfamiliar APKs, (Tiwari, S. R., et al. 2018) used a combination of sensitive permissions and API attributes and an ensemble-learning model based on decision trees and KNN classifiers. In addition, (Zhang, H., et al. 2019) developed behavioral semantics by estimating the confidence of association rules between abstracted API calls used to construct an application and then identifying them using various machine learning approaches (KNN, RF, SVM). Using the same features and machine learning algorithms as MaMaDroid (Mariconti, E., et al. 2017), the proposed model outperformed it.

(Suarez-Tangil, G., et al. 2017) proposed DroidSieve as a lightweight approach for detecting disguised Android malware. They divided an application's features into two high-level categories: syntactic (code) features generated from the application's source code, and metadata and resource-centric characteristics resource-centric features are derived from the assets. These characteristics are utilized for training an extra trees model, which has a high

accuracy of 99 % in predicting the label of a particular application. The number of malware used in this study is 16,141 apps. In (Battista, P., et al. 2016), the authors proposed an approach to localize malicious behavior at a finer grain, i.e., at the payload level. They used model checking to test their model against two of the most diffused malware families in the Android environment: the DroidKungFu and the Opfake families. Also, they implemented a prototype tool and conducted experiments to prove of the concept of their methodology. Finally, they obtained an accuracy ranging from 0.83% to 0.94%. DroidCat is the first unified dynamic malware detection technique (Cai, H., et al. 2019) used to detect Android malware and pinpoint its malware family. It uses supervised machine learning to train a multi-class classifier utilizing a variety of benign app and malware behavioral patterns. In contrast to previous heuristics-based machine learning approaches, DroidCat's feature set is only determined through a thorough dynamic characterization study of benign and malicious apps. All distinguishing features reveal behavioral variations between benign and malicious apps. DroidCat was tested using leave-one-out cross-validation with 136 benign and 135 malicious apps, and the accuracy reached 92%.

Apposcopy is another static analysis approach for detecting malware in the mobile app ecosystem (Feng, Y., et al. 2014). Apposcopy does a comprehensive static analysis of Android applications to extract data-flow and control-flow properties and then utilizes the results to determine whether the app belongs to a known malware family. It depends on the concept that malware belonging to the same family exhibits a similar set of behaviors, which an auditor can record using Apposcopy's Datalog-based malware definition language. One more hybrid analysis-based process approach is introduced by (Ding, C., et al. 2021) to detect and classify Android malware, improving static and dynamic network traffic properties. Permissions and intent are utilized as input static features in the detection layer. After feature selection and comparing their performance on different algorithms, the best static detection algorithm and feature selection method, namely random forest, and chi-square test are determined. According to the tests, Apposcopy can detect malware with high accuracy of 90 %. The claimed detection rate was 95.04% at the end.

A few techniques have been proposed by (Shao, K., et al. 2021) and (XU, Z., et al. 2019) to improve the performance of android malware and family categorization techniques. A new malware family-based bagging ensemble method (FB2) and an Android malware detection scheme (FB2Dorid) is introduced (Shao, K., et al. 2021). The first step was to decompile APK files to extract the features. The relief feature selection technique was then used to choose a subset of the most significant features. The bagging integration algorithm's sampling method is then improved using two sampling strategies: equal amount sampling and family information sampling. Finally, the base classifiers and FB2 were integrated. The authors devised a series of tests to test the FB2 scheme's viability, where the testing findings show that the proposed FB2 system performed well, with a claimed 97% accuracy rate. In (XU, Z.,

et al. 2019), the authors developed an Android malware family classification and characterization approach based on control flow graph (CFG) and data flow graph (DFG) features. They conducted some intriguing tests to evaluate the proposed strategy. They discovered through trials that the family classification model using a horizontal mix of CFG and DFG as features works the best and that their family classification model outperforms CDGDroid, Drebin, and the majority of antivirus tools gathered in VirusTotal. The claimed accuracy achievement of 94.7%.

Our proposal differs in how it handles the malware and its family differently. It utilizes different approaches to get the best possible results. The details of the proposed approach are described in the following section.

Methodology

Machine Learning Process

Machine Learning (Machine learning, e., et al. 2022) is a method of analyzing data using software tools (algorithms) to develop a model that aids in patterns discovery of regularities in datasets. It teaches machines to learn from previous experiences (data) to predict future events or data instances. Feature vectors are crucial components of machine learning, and they are often created for the specific task associated with a particular algorithm. Machine learning is based on the concept of obtaining the probability data distribution. Machine learning has to have a training dataset to learn from, gaining some knowledge from it.

The machine model is deployed to accept new data and produce an appropriate output, as shown in Figure 1. This output is frequently a prediction or classification of some type. The selection of the machine learning technique depends on what kind of data to predict. Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning are the four primary types of machine learning. When it comes to supervised learning, algorithms like K-Nearest Neighbor (KNN) are trained on labeled data and could be used. In contrast, unsupervised learning includes algorithms such as K-Means that could be trained on unlabeled data. Semi-supervised learning consists of a mix of the two preceding types. For example, reinforcement learning teaches a machine to complete a multi-step process with clearly defined rules. In this paper, supervised learning is applied to classify a labeled dataset.

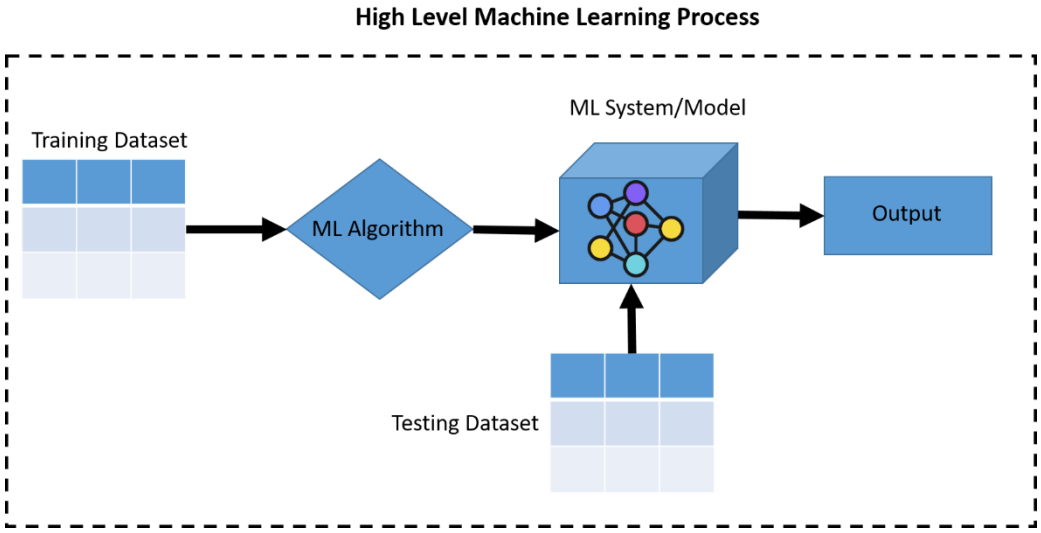

Figure 1. Machine Learning Process

The Proposed Approach

This subsection proposes an effective systematic and functional parallel machine-learning model for dynamically detecting and classifying Android malware categories and families. As shown in Fig. 2, the proposed model consists of the following phases:

1- Data Pre-Processing Phase

This phase handles most of the required processes in which the following stages are conducted:

- **Data processing**: The following process is performed on the data set, (i) the NaN is removed. (ii) Duplicated instances are deleted. (iii) Features normalization is applied using the MinMax scaling for feature normalization since the given dataset has minimal variance and ambiguity. The rescaling of real-valued numeric characteristics to a defined range is normalization (e.g., 0 and 1). When using a model that relies on the magnitude of values, scaling the input characteristics is critical. MinMax scaling normalizes data, as shown in (1).

$$X_{norm} = \frac{X_i - X_{min}}{X_{max} - X_{min}} \quad (1)$$

where X_i , X_{min}, and X_{max} are the original value of the feature, minimum, and maximum values, respectively

- **Feature selection**: The most significant features for malware category and family classification are selected in this phase. The standard feature selection methods are used, and their effect on the system performance is evaluated. Based on the evaluation results, a suitable method is selected and applied.

2- Malware category and malware family classification (Training Phase): In this stage, two parallel classifiers are implemented to classify both malware category and malware family. The most common classifiers are used in the proposed model. In addition, the performance of each classifier is recorded and compared with other classifiers. The best classifier for malware category and family classification is selected and applied based on the evaluation results.

3- Testing Phase: In this phase, the proposed model is tested by entering a specific android malware. The malware category and family are predicted in parallel; the obtained results are compared with other models.

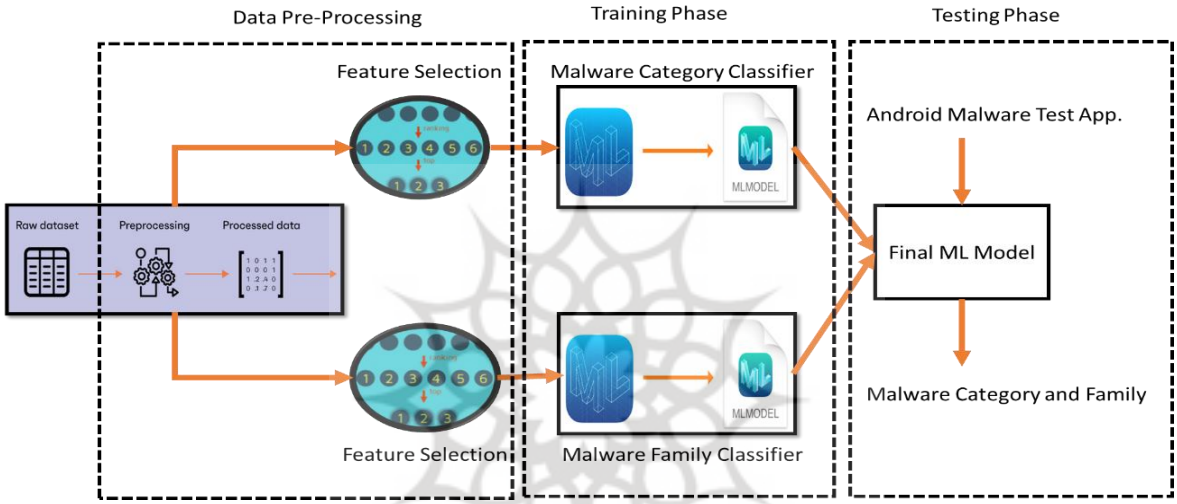

Figure 2. Graphical representation of the proposed model

As shown in Figure 2, the proposed approach utilizes a parallel execution to the Android malware category and Android malware family classification. This approach applies multiple machine learning algorithms simultaneously, and the best algorithm(s) is selected at the testing phase.

Results

Evaluation Metrics

In this paper, the following evaluation metrics are used to analyze the performance of the proposed model. Accuracy, recall, False Positive Ratio (FPR), and False Negative Ratio (FNR) are computed using the given below equations based on the well-known parameters TP (true positive), FP (false positive), TN (true negative), and FN (false negative).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$FPR = \frac{FP}{FP+TN} = 1 - TNR \quad (4)$$

$$FNR = \frac{FN}{TP+FN} = 1 - TPR \quad (5)$$

Accuracy is the ratio of the number of correct predictions to the total number of input samples.

Recall: It is the number of correct positive results divided by all relevant samples (all samples that should have been identified as positive).

False Positive Ratio (FPR): The proportion of the data points with a known negative condition for which the test result is positive. This rate is sometimes called the fall-out.

False Negative Ratio (FNR): The proportion of the data points with a known positive condition for which the test result is negative. This rate is sometimes called the miss rate.

CCCS-CIC-AndMal-2020 Dataset

In this paper, a CCCS-CIC-AndMal-2020 dataset is used for malware category and family classification. CCCS-CIC-AndMal-2020 is a recently released Android Malware Dataset (Gagnon, F., et al. 2022). It comprises 400K android apps (200K benign and 200K malware). It includes static elements such as Android malware families, permissions, intents, and dynamic features like API calls and its generated log files. Process logs, packages, log states, battery states, and other collected information are also included in the dataset. It contains 14 malware categories, as shown in Table 1 (IT World Canada, et al. 2022) (Kaspersky, et al. 2022) and 180 Families.

The dataset is divided into two main parts: the static and dynamic parts; the static part focuses on analyzing Android apps as benign or malware in general, while the dynamic part focuses on analyzing Android Malware apps only. The Dynamic part is divided into two subparts:

i. Data dynamically collected before rebooting the Android emulator: It contains 28,380 Android Malware apps, as shown in Table 2, which describes the number of families in each category, the number of malware apps of each category, and a list of famous families in each category.

ii. Data dynamically collected after rebooting the Android emulator: includes 25,059 Android Malware apps. Because the number of applications in the first part is more significant than the second part, our research utilizes the first part (before reboot). The second part (after reboot) will be considered in our future work.

Table 1. description of android Malware Categories

#	Malware Category	Malware Definition
1	Adware	It's a malicious program that displays intrusive adverts on the user's screen, mainly when using web services
2	Backdoor	Backdoors are unnoticed entrances into a smartphone. Backdoors, in other words, are a means to circumvent a smartphone's authentication and elevate privileges, enabling an attacker to access the device at any time
3	File Infector	a file infector is a type of malware that infects APK files. Android Package Kit (APK) is a file that contains all of an application's data. With the help of APK files, the file infector is set up. After that, when APK files are installed, the virus is run.
4	No_Category	New programs with malicious activities and not classified to any popular category
5	PUA	PUAs are potentially undesirable programs that come bundled with genuine, free-to-install software. They're sometimes referred to as potentially unwanted programs (PUPs). PUAs aren't always dangerous, and it all depends on how they'll be utilized. This sort of malware includes adware, spyware, and hijackers.
6	Ransomware	Ransomware is a type of software that encrypts files and folders on a machine so that users can't access them. It wants a large ransom in exchange for the decryption key to allow the data to be accessed.
7	Riskware	Riskware is a genuine software that threatens the device's security vulnerabilities. Even though it is legitimate software, it collects data from the device and leads users to malicious websites.
8	Scareware	Scareware is a fear inducer that encourages users to download or purchase harmful applications by instilling dread in their brains—for example, persuading people to install phony software that claims to protect the device.
9	Trojan	Trojans are nefarious imposters that pose as simple programs. They can steal data from the device while remaining undetected in the background.
10	Trojan_Banker	Banker programs are intended to steal your account information for online banking, e-payment, and credit or debit cards.
11	Trojan_Dropper	Hackers use these programs to install Trojans or viruses to prevent dangerous programs from being detected. Not all antivirus systems can scan all of the components contained within this sort of Trojan.
12	Trojan_SMS	If you use your mobile device to send text messages to premium rate phone numbers, These programs can cost you money.
13	Trojan_Spy	Programs can follow the data you type via your keyboard, take screenshots, and access a list of running applications to spy on how you use your phone.
14	Zero_Day	New programs that Anti-Malware solutions have failed to detect.

Here are six characteristics extracted to understand the behavioral changes of these Android malware categories and families. The following are the significant characteristics that were extracted:

- Memory Features: It defines the activities performed by malware utilizing memory. (23 Features)
- API: The Application Programming Interface (API) features delineate the communication between two applications. (105 Features)
- Network: The network features describe the data transmitted and received between other devices in the network. It indicates foreground and background network usage. (4 Features)
- Battery: Battery features describe malware's access to battery wakelock and services. (2 Features)

- Logcat: Logcat features write log messages corresponding to a function performed by malware. (6 Features)
- Process: Process features count malware's interaction with a total number of processes. (1 Feature)

As can be seen, the total number of extracted features from the six characteristics is 141 features.

Table 2. Dynamic Analysis (before reboot) Dataset details

#	Malware Category	No. of Malware Families	No. of Samples	Common Malware Family
1	Adware	43	5838	gexin, batmobi, ewind, shedun, pandaad, appad, dianjin, gmobi, hummingbird, mobisec, loki, kyhub, and adcolony
2	Backdoor	11	591	mobby, kapuser, hiddad, dendroid, levida, fobus, moavt, androrat, kmin, pyls, and droidkungfu
3	File Infector	5	129	leech, tachi, commplat, gudex, and aqplay
4	No_Category	1	1048	No_Category
5	PUA	9	665	apptrack, secapk, wiyun, youmi, scamapp, utchi, cauly, and umpay
6	Ransomware	8	1861	congur, masnu, fusob, jisut, koler, lockscreen, slocker, and smsspy
7	Riskware	19	7261	badpac, mobilepay, wificrack, triada, skymobi, deng, jiagu, smspay, smsreg, and tordow
8	Scareware	4	462	avpass, mobwin, and fakeapp
9	Trojan	38	4412	gluper, lotoor, rootnik, guerrilla, gugi, hqwar, obtes, and hypay
10	Trojan_Banker	11	118	minimob, marcher, faketoken, zitmo, bankbot
11	Trojan_Dropper	9	837	Ramnit, cnzz, rooter, gorpo, xiny
12	Trojan_SMS	10	1028	Opfake, plankton, boxer, vietsms
13	Trojan_Spy	11	1801	Smsthief, qqspy, spyagent, smforw
14	Zero_Day	1	2329	Zero_Day
	Sum	180	28,380	

Feature Selection

As mentioned above, the total extracted features from the six types of characteristics are 141 features. However, removing irrelevant and insignificant features is necessary to improve the classification performance. This section applies two common feature selection methods, Chi2 and Mutual Information (MI), for malware category detection and malware family detection.

Feature Selection for Malware Category Detection Performance

In this subsection, a set of experiments has been performed to compare the Chi2 and Mutual Information (MI) feature selection methods for detecting the android malware category. The two approaches are used with four standard classifiers (Random Forest (RF), K-Nearest

Neighbors (KNN), J48, and Naive Bayes (NB)). Table 3 shows the obtained accuracies for classifying malware categories using Chi2 and MI feature selection methods, respectively, with the selected four classifiers RF, KNN, J48, and NB for different threshold values 20%, 40%, 60%, 80%, and 100%. Also, the obtained data is plotted in Figure 3. After comparing the results of applying both Chi2 and MI feature selection methods for malware category detection, it was found that the best feature selection method is MI at a threshold of 60%.

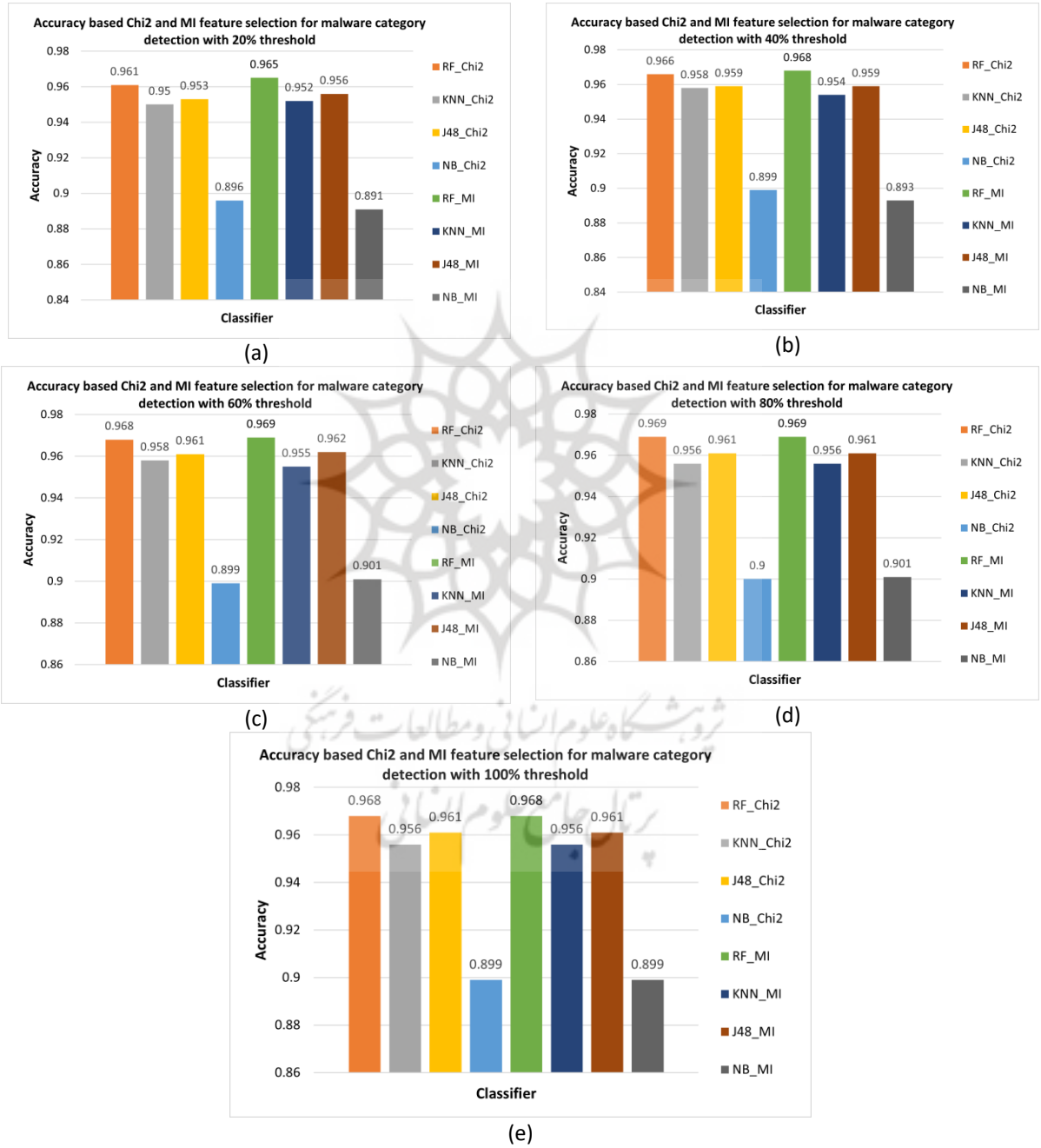

Figure 3. Obtained accuracy based Chi2 and MI feature selection for malware category detection, (a) 20% threshold, (b) 40% threshold, (c) 60% threshold, (d) 80% threshold, (e) 100% threshold

As given in Figure 3, on average, as the threshold increases, the accuracy is increased. However, the best performance gained was by RF_Chi2, where 80% and 100% thresholds are used. Similarly, RF_MI seems to perform the best when 60% and 100% thresholds are used. On the other hand, the worst performance was given by NB_Chi2.

Feature Selection for Malware Family Detection Performance

In this subsection, experiments have been performed to compare the Chi2 and MI feature selection methods for detecting the android malware family. The two approaches are used with four standard classifiers (RF, KNN, J48, and NB). Table 4 shows the obtained accuracies for classifying malware families using Chi2 and MI feature selection methods, respectively, with the selected four classifiers RF, KNN, J48, and NB for different threshold values 20%, 40%, 60%, 80%, and 100%. Also, the obtained data is plotted in Figure 4. After comparing the results obtained while applying both Chi2 and MI feature selection methods, it was found that the best feature selection method is Chi2 at the threshold of 80%.

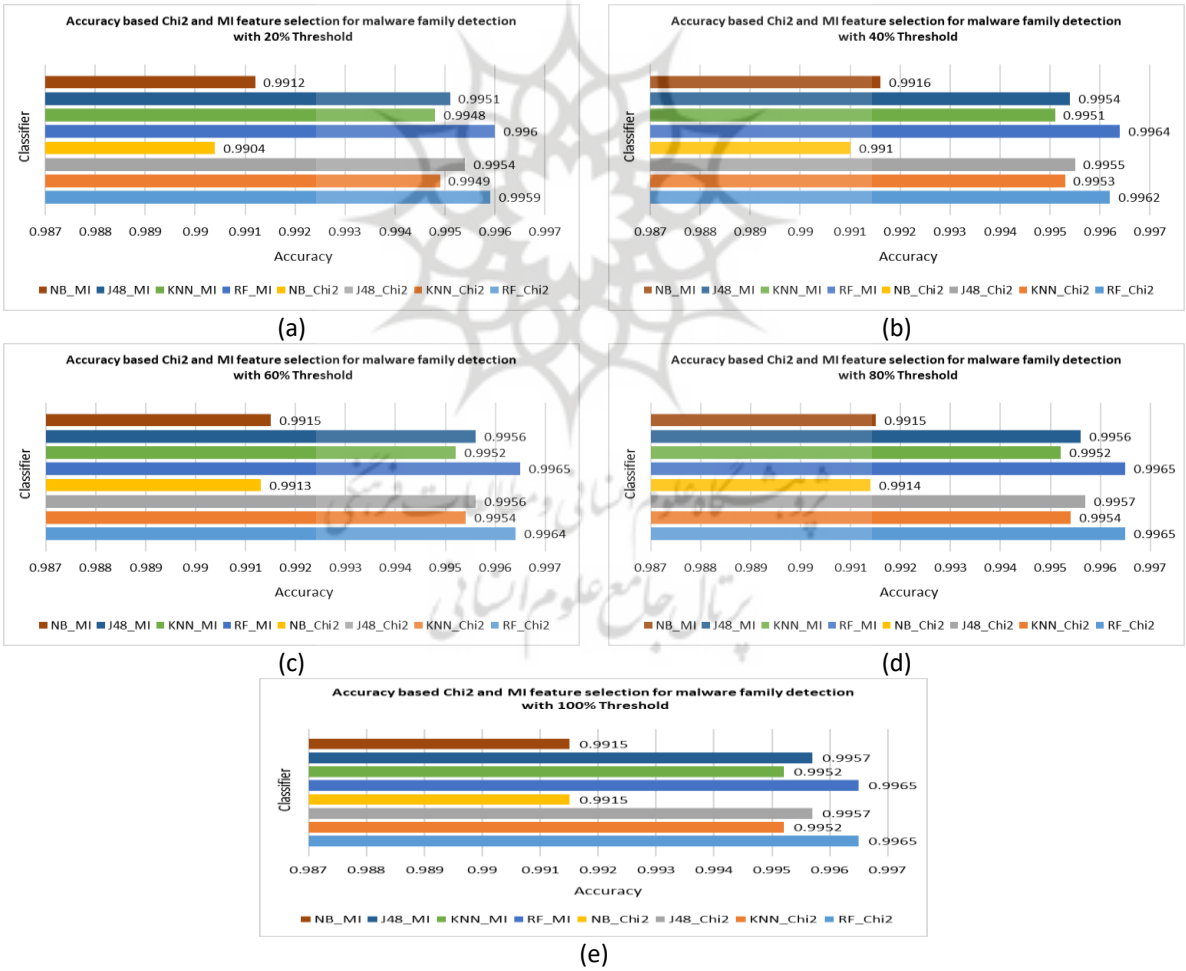

Figure 4. Obtained accuracy based Chi2 and MI feature selection for malware family detection, (a) 20% threshold, (b) 40% threshold, (c) 60% threshold, (d) 80% threshold, (e) 100% threshold

Parallel AI Detection Model Experiments

In this section, a set of experiments have been performed to study the performance of the proposed model. Each experiment is performed ten times, and the average reading of the obtained results is recorded. The computing environment details are mentioned in Table 5. Table 6 summarizes used parameters in each model malware category and malware family.

Table 5. Computing Environment

Parameter	Value
Operating System	MacOS High Sierra v 10.13.4
CPU	2.8 GHz Intel Core i7
RAM	16 GB
Python Version	2.7

Table 6. Experiments parameters

Parameter	Malware Category Model	Malware Family Model
Feature Selection Method	MI	Chi2
Feature Selection Threshold	60%	80%
Training and Testing Percentage	Cross-Validation (K=10)	Cross-Validation (K=10)
Number of Classifiers	8	8
Types of Classifiers	RF, KNN, J48, NB, MLP, SVM, AB and LR	RF, KNN, J48, NB, MLP, SVM, AB and LR

Malware category and malware family Classification Performance

The following machine learning algorithms are used to evaluate and compare the effectiveness of the proposed approach: J48, NB, Support Vector Machine (SVM), AdaBoost (AB), Logistic Regression (LR), KNN, RF, and Multilayer Perceptron (MLP). The results of applying machine learning classifiers to classify the malware category model are shown in Table 7 and Figure 5. It has been found that the RF classifier achieves the best accuracy with 96.89%, a recall ratio of 66.46%, an FPR of 1.85%, and FNR of 33.54%, and a training time is approximately 987 seconds. After that, the MLP classifier is the second one with an accuracy of 96.32%, a recall ratio of 58.48%, an FPR of 2.17%, FNR of 41.62%, and the training time is close to 40874 seconds. Then J48 classifier is occupied the third rank with an accuracy of 96.16%, a recall ratio of 64.45%, an FPR of 2.22%, and FNR of 35.55%, and the training time is reached 1389 seconds. Finally, the NB classifier is the last ranked one with an accuracy of 90.15%, a recall ratio of 31.95%, an FPR of 5.12%, and an FNR of 68.05%. However, it has the best training time with 189 seconds.

Table 7. ML Results for Malware Category Classification

Classifier	Time (seconds)
RF	987.16
KNN	1810.58
J48	1389.67
NB	189.94
MLP	40874.48
SVM	23526.07
AB	3178.18
LR	1599.46

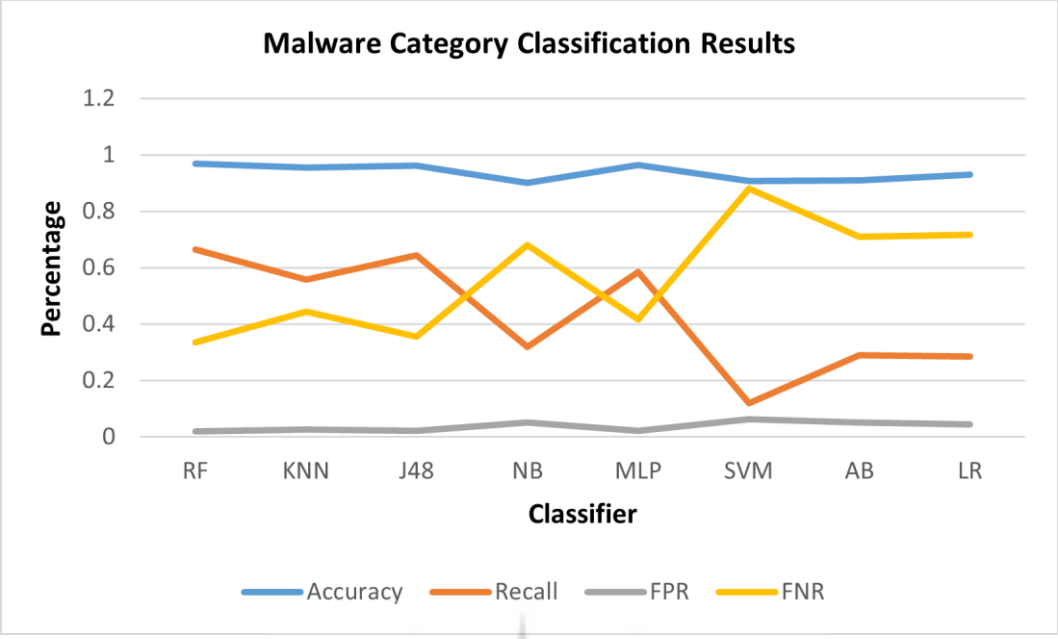

Figure 5. Malware Category Classification Results

The machine learning classifiers results in the malware family model are shown in Table 8. We found that the RF classifier achieves the best accuracy with 99.65%, a recall ratio of 36.27%, an FPR of 0.18%, and an FNR of 63.73%, and the training time is approximately 312 seconds. After that, the MLP classifier comes in the second rank with an accuracy of 99.60%, recall ratio of 32.64%, FPR of 0.21%, FNR of 67.36%, and the training time is close to 63165 seconds. The J48 classifier ranked third with an accuracy of 99.57%, a recall ratio of 34.21%, FPR of 0.22%, FNR of 65.79%, and the training time was 470 seconds. Lastly, the AB classifier results with an accuracy of 99.05%, recall ratio of 1.87%, FPR of 0.56%, FNR of 98.13%, and a training time close to 6706 seconds. However, according to training time, the best classifier is NB, which was completed in 113 seconds only and achieved an accuracy of 99.14%, recall ratio of 28.87%, FPR of 0.43%, and FNR of 71.13%.

Table 8. ML Results for Malware Family Classification

Classifier	Accuracy	Recall	FPR	FNR	Time (seconds)
RF	0.9965	0.3627	0.0018	0.6373	312.39
KNN	0.9954	0.2697	0.0024	0.7303	922.07
J48	0.9957	0.3421	0.0022	0.6579	470.99
NB	0.9914	0.2887	0.0043	0.7113	113.49
MLP	0.9960	0.3264	0.0021	0.6736	63165.61
SVM	0.9911	0.0262	0.0052	0.9738	23917.79
AB	0.9905	0.0187	0.0056	0.9813	6706.97
LR	0.9933	0.0844	0.0037	0.9156	6251.64

Related Work Performance Comparison

Finally, the obtained results are compared with the experimental results of other work as in Tables 9 and 10. it is found that the proposed model is the best for many reasons. It is applied to a large dataset, the latest version of the dataset. It achieves the highest accuracy among other research studies that classify Android malware and Android Malware families. Using an RF classifier, we found that the proposed model achieved the best accuracy, which reached over 96% compared to (Taheri, L., et al. 2019), (Imtiaz, S. I., et al. 2021), (Abuthawabeh, M., et al. 2019), and (Abuthawabeh, M., et al. 2020) with accuracies of 83.3%, 82.2%, 80.2%, and 79.97%, respectively. The work of (Taheri, L., et al. 2019) was based on the same classifier as our proposal, while (Imtiaz, S. I., et al. 2021) used Deep ANN. Also, (Abuthawabeh, M., et al. 2019) was based their work on the RF classifier while (Abuthawabeh, M., et al. 2020) used the Extra Trees classifier.

Table 9. Result comparison of malware category classification

Dataset	Accuracy
(Taheri, L., et al. 2019)	83.3% (RF)
(Abuthawabeh, M., et al. 2020)	79.97% (Extra Trees)
(Abuthawabeh, M., et al. 2019)	80.2% (RF)
(Imtiaz, S. I., et al. 2021)	82.2% (Deep ANN)
Our Model	96.89% (RF)

The results of the malware family between our model and other works are shown in Table 10. It has been found that the proposed model achieved the best accuracy, with a percentage over 99% using an RF classifier. Then, the authors (Abuthawabeh, M., et al. 2020) achieved an accuracy of close to 67% by using the Extra Trees classifier. But the approach proposed in (Imtiaz, S. I., et al. 2021) achieved an accuracy of almost 65% by using an NB classifier. On the other hand, (Taheri, L., et al. 2019) performed the worst accuracy, which is close to 59.7%, using an RF classifier.

Table 10. Result comparison of malware family classification

Dataset	Accuracy
(Taheri, L., et al. 2019)	59.7% (RF)
(Abuthawabeh, M., et al. 2020)	66.71 % (Extra Trees)
(Imtiaz, S. I., et al. 2021)	65% (NB)
Our Model	99.65% (RF)

Discussion

As can be seen in this paper, the problem of Android malware detection is one of the important problems due to the widespread usage of Android in millions of limited resources and/or less secure devices. Therefore, the detection of Android malware is a challenge. In addition, detection of malware category and family became an issue for countermeasures.

Although many techniques are already proposed for malware detection, there is a lack of accuracy, detection of the malware family, or even the current approaches take too much time, making them unsuitable for runtime detection. Therefore, the proposed approach utilizes the parallel detection concept for the android malware family and android malware category. The proposed model consists of two main phases (i) data preprocessing: it is applied first to clean, formally represent the data, and extract efficient data features. (ii) data classification: the proposed framework is a generic framework that could merge similar or different techniques for malware categorization and family detection, respectively.

After experimenting with many of the algorithms on one of the most extensive datasets, it turns out that the proposed approach could enhance the accuracy and required time for malware detection. It reached an accuracy of over 96% for malware category classification and over 99% for malware family classification. This enables the proposed approach utilization in the runtime and dynamic detection. Also, the work done in this paper shows the importance of artificial intelligence techniques in Android malware classification. However, the proposed framework has some limitations as follows:

- The proposed approach is based on a dynamic analysis of android apps.
- Static features could add more depth if applied before dynamic analysis.

Conclusion

Android malware is one of the most dangerous threats on the Internet, and its prevalence has increased dramatically in recent years. There are various machine learning-based approaches for detecting and classifying Android malware. This article offers a Machine Learning model that uses feature selection and a Machine Learning Classifier to perform malware classification and characterization successfully. Our model has shown promising results, with malware category classification accuracy of over 96 % and malware family classification accuracy of over 99 %. Furthermore, our algorithm accurately classifies most of the occurrences of 180 malware families and 14 malware categories, demonstrating efficiency. Because it can categorize a more extensive dataset and malware families, our Machine Learning Model is scalable. It allows for multi-class characterization with high precision and a low rate of false positives. In the future, we plan to provide an online service that allows users to check if a program is a malware or not before downloading it and determine its category and family. This measure would go a long way toward ensuring the security of an Android smartphone.

Conflict of interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data

fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article

References

Abuthawabeh, M. K., & Mahmoud, K. W. (2019). Android malware detection and categorization based on conversation-level network traffic features. 2019 International Arab Conference on Information Technology (ACIT). https://doi.org/10.1109/acit47987.2019.8991114

Abuthawabeh, M., & Mahmoud, K. (2020). Enhanced Android malware detection and family classification, using conversation-level network traffic features. The International Arab Journal of Information Technology, 17(4A), 607-614. https://doi.org/10.34028/iajit/17/4a/4

Arp, D., Spreitzenbarth, M., Hübner, M., Gascon, H., & Rieck, K. (2014). Drebin: Effective and explainable detection of Android malware in your pocket. Proceedings 2014 Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2014.23247.

Arslan, R. S., Doğru, İ. A., & Barişçi, N. (2019). Permission-based malware detection system for Android using machine learning techniques. International Journal of Software Engineering and Knowledge Engineering, 29(01), 43-61. https://doi.org/10.1142/s0218194019500037

Battista, P., Mercaldo, F., Nardone, V., Santone, A., & Visaggio, C. A. (2016). Identification of Android malware families with model checking. Proceedings of the 2nd International Conference on Information Systems Security and Privacy. https://doi.org/10.5220/0005809205420547

Cai, H., Meng, N., Ryder, B., & Yao, D. (2019). DroidCat: Effective Android malware detection and categorization via app-level profiling. IEEE Transactions on Information Forensics and Security, 14(6), 1455-1470. https://doi.org/10.1109/tifs.2018.2879302

Ding, C., Luktarhan, N., Lu, B., & Zhang, W. (2021). A hybrid analysis-based approach to Android malware family classification. Entropy, 23(8), 1009. https://doi.org/10.3390/e23081009

El Fiky, A. H., El Shenawy, A. & Madkour, M. A. (2021). A Survey of Malware Detection Techniques for Android Devices. AL-AZHAR ENGINEERING FIFTEETHEN INTERNATIONAL CONFERENCE

El Fiky, A. H., El Shenawy, A. & Madkour, M. A. (2021). Detection of Android Malware using Machine Learning. International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC) DOI: 10.1109/MIUCC52538.2021.9447661

Faghihi, F., Abadi, M., & Tajoddin, A. (2018). SMSBotHunter: A novel anomaly detection technique to detect SMS Botnets. 2018 15th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC). https://doi.org/10.1109/iscisc.2018.8546934.

Fan, M., Luo, X., Liu, J., Wang, M., Nong, C., Zheng, Q., & Liu, T. (2019). Graph embedding based familial analysis of Android malware using unsupervised learning. 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). https://doi.org/10.1109/icse.2019.00085

Feizollah, A., Anuar, N. B., Salleh, R., Suarez-Tangil, G., & Furnell, S. (2017). AndroDialysis: Analysis of Android intent effectiveness in malware detection. Computers & Security, 65, 121-134. https://doi.org/10.1016/j.cose.2016.11.007

Feng, Y., Anand, S., Dillig, I., & Aiken, A. (2014). Apposcopy: Semantics-based detection of Android malware through static analysis. Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. https://doi.org/10.1145/2635868.2635869

Gagnon, F. and Massicotte, F., 2022. Revisiting Static Analysis of Android Malware. [online] Usenix.org. Available at: <https://www.usenix.org/conference/cset17/workshop-program/presentation/gagnon> [Accessed 29 February 2022].

Ideses, I., & Neuberger, A. (2014). Adware detection and privacy control in mobile devices. 2014 IEEE 28th Convention of Electrical & Electronics Engineers in Israel (IEEEI). https://doi.org/10.1109/eeei.2014.7005849

Imtiaz, S. I., Rehman, S. U., Javed, A. R., Jalil, Z., Liu, X., & Alnumay, W. S. (2021). DeepAMD: Detection and identification of Android malware using high-efficient deep artificial neural network. Future Generation Computer Systems, 115, 844-856. https://doi.org/10.1016/j.future.2020.10.008

IT World Canada - Information Technology news on products, services and issues for CIOs, IT managers and network admins. 2022. Understanding Android Malware Families (UAMF) – The Foundations (Article 1) - IT World Canada. [online] Available at: <https://www.itworldcanada.com/blog/understanding-android-malware-families-uamf-the-foundations-article-1/441562> [Accessed 1 March 2022].

Kaspersky. 2022. What is a Trojan horse and what damage can it do?. [online] Available at: <https://www.kaspersky.com/resource-center/threats/trojans> [Accessed 17 March 2022].

Ko, J., Jo, J., Kim, D., Choi, S., & Kwak, J. (2019). Real time Android ransomware detection by analyzed Android applications. 2019 International Conference on Electronics, Information, and Communication (ICEIC). https://doi.org/10.23919/elinfocom.2019.8706349.

Lou, S., Cheng, S., Huang, J., & Jiang, F. (2019). TFDroid: Android malware detection by topics and sensitive data flows using machine learning techniques. 2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT). https://doi.org/10.1109/infoct.2019.8711179

Lueg, C., 2022. Cyber attacks on Android devices on the rise. [online] Gdatasoftware.com. Available at: <https://www.gdatasoftware.com/blog/2018/11/31255-cyber-attacks-on-android-devices-on-the-rise> [Accessed 18 January 2022].

Ma, Z., Ge, H., Liu, Y., Zhao, M., & Ma, J. (2019). A combination method for Android malware detection based on control flow graphs and machine learning algorithms. IEEE Access, 7, 21235-21245. https://doi.org/10.1109/access.2019.2896003

Machine learning, e., 2022. Machine learning, explained | MIT Sloan. [online] Best Online Course. Available at: < https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained> [Accessed 17 February 2022].

Mariconti, E., Onwuzurike, L., Andriotis, P., De Cristofaro, E., Ross, G., & Stringhini, G. (2017). MaMaDroid: Detecting Android malware by building Markov chains of behavioral models. Proceedings 2017 Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2017.23353

Mylonas, A., & Gritzalis, D. (2012). Practical malware analysis: The hands-on guide to dissecting malicious software. Computers & Security, 31(6), 802-803. https://doi.org/10.1016/j.cose.2012.05.004.

Shao, K., Xiong, Q., & Cai, Z. (2021). FB2Droid: A novel malware family-based bagging algorithm for Android malware detection. Security and Communication Networks, 2021, 1-13. https://doi.org/10.1155/2021/6642252

StatCounter Global Stats. 2022. Operating System Market Share Worldwide | Statcounter Global Stats. [online] Available at: <https://gs.statcounter.com/os-market-share> [Accessed 17 January 2022].

Statista. 2022. Android vs iOS market share 2023 | Statista. [online] Available at: <https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/> [Accessed 17 January 2022].

Suarez-Tangil, G., Dash, S. K., Ahmadi, M., Kinder, J., Giacinto, G., & Cavallaro, L. (2017). DroidSieve. Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy. https://doi.org/10.1145/3029806.3029825

Taheri, L., Kadir, A. F., & Lashkari, A. H. (2019). Extensible Android malware detection and family classification using network-flows and API-calls. 2019 International Carnahan Conference on Security Technology (ICCST). https://doi.org/10.1109/ccst.2019.8888430.

Tiwari, S. R., & Shukla, R. U. (2018). An Android malware detection technique based on optimized permissions and API. 2018 International Conference on Inventive Research in Computing Applications (ICIRCA). https://doi.org/10.1109/icirca.2018.8597225

XU, Z., Ren, K., & Song, F. (2019). Android malware family classification and characterization using CFG and DFG. 2019 International Symposium on Theoretical Aspects of Software Engineering (TASE). https://doi.org/10.1109/tase.2019.00-20

Yerima, S. Y., & Khan, S. (2019). Longitudinal performance analysis of machine learning based Android malware detectors. 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security). https://doi.org/10.1109/cybersecpods.2019.8885384

Zhang, H., Luo, S., Zhang, Y., & Pan, L. (2019). An efficient Android malware detection system based on method-level behavioral semantic analysis. IEEE Access, 7, 69246-69256. https://doi.org/10.1109/access.2019.2919796

Bibliographic information of this paper for citing:

El Fiky, Ahmed Hashem; Madkour, Mohamed Ashraf & El Shenawy, Ayman (2022). Android Malware Category and Family Identification Using Parallel Machine Learning. *Journal of Information Technology Management,* 14 (4), 19-39. https://doi.org/ 10.22059/jitm.2022.88133