Journal of Theoretical and Applied Information Technology
31st October 2019. Vol.97. No 20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MALDROID: ATTRIBUTE SELECTION ANALYSIS FOR MALWARE CLASSIFICATION

[1]RAHIWAN NAZAR ROMLI, [1]MOHAMAD FADLI ZOLKIPLI, [1]MOHD ZAMRI OSMAN

[1]Faculty of Computer Systems & Software Engineering University Malaysia Pahang, 26200 Gambang, Malaysia

E-mail: [1]rahiwan@ump.edu.my, [1]fadli@ump.edu.my, [1]zamriosman@ump.edu.my

ABSTRACT

Android is the most dominant operating system in the mobile market and the number of Android users is increasing year by year. Malware authors use android market as a hub for malicious apps and spread malware to users with the intention to threaten privacy; and this has remained undetected due to the weakness in signature-based detection. A major problem with malware detection is the existence of numerous features in malware code and the need to look at the relevant features in malware analysis. As a result, applying any security solution in malware analysis is considered inefficient because mobile devices have limited resources in terms of its memory, processor and storage. Hence, the objective of this paper is to find the most effective and efficient attribute selection and classification algorithm in malware detection. Moreover, in order to get the best combination between attribute selection and classification algorithm, eight attributes selection and seven categories machine learning algorithm are applied in this study. The experiment evaluated 8000 real data samples and the result showed that InfoGainEval and KNN algorithm are the most selected in attribute selection and classification process.

Keywords: *Android, Malware, Malware Analysis, Machine Learning Algorithm, Info Gain Evaluation*

1. INTRODUCTION

Android is an open source platform. It offers performance, credibility, easiness to customize and is easy to download by the users; either free or charged. According to [1] there are more than 2 billion devices around the world across different types of devices such as smartphone, automobile, TV and more that are using Android. With the increase in android users, it has also contributed to the rise and upgraded capabilities of malware attack. Android OS is very popular among the younger generation as they use their mobile devices for web browsing, downloading and playing games, personal health, banking etc [2]. All these activities attract malware attackers whose intention is to steal their personal information, collapse their mobile system, sending premium service SMS; that would result in the victims to suffer much graver consequences.

Android OS offers many categories of applications in its market store which consist of games, health management, banking, navigation and many more. With the steady growth of the market store, it offers users the convenience to download and install any applications that they want. This created a flooding of applications in the android market; applications from principal market stores such as Google Store and Play store that offer legal application, while the third party market such as China market contain a lot of application that are not inspected and labelled for their security which contribute the proliferation of malware [3].

In malware detection, the features in android application consist of many attributes. Selecting the relevant attributes is complicated due to the need for the security analyst to inspect the application in order to distinguish between a malware or benign based on the characters or elements that are frequently used by malware or benign applications. In malware detection, static analysis is done by reverse engineering, scanning and identifying attributes in application. The problem becomes bigger when we try to extract the unwanted attributes due to exceeding the suitable amount or too many features. For example, [4] claim that anomaly based detection needs only minimal feature for better results in malware classification. Moreover, it also lead to some drawbacks to machine learning classification such as the deviation of learning algorithm, over-fitting, reduced generality and model run time being

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

increased [5].However, collecting and using the large number of attributes also contribute to increasing consumption of system overhead such as storage, memory, CPU and power.

In consideration to the emergence of new malware apps, there is a need for a new model or system that may operate more effectively and efficiently in identifying malware. A fine attribute selection in data pre-processing will lead to classification accuracy and low system usage. Therefore, this work tries to search attribute that leads to a more accurate detection malware as a whole.

Choosing specific attribute selection and method for malware detection that is related to machine learning algorithm is challenging. In order to solve this issue, this paper analyzed eight sets of Attribute Selection and method and six classifier algorithms from seven categories algorithms, specifically on machine learning, in order to find the best combination which contributes to the highest accuracy in malware detection. The result and methods used will be discussed in next section.

The rest of the paper is organized as follows. The next section presents the related works. The methodology is illustrated in Section 3; followed by Section 4 that explains the experimental results. Finally, Section 5 presents the conclusion and further research agenda.

2. RELATED WORK

In recent years, there has been an increasing amount of literature on using of various classification and attribute selection in malware detection. In this section, we will present previous researches in malware classification in the perspective of selecting Android feature, classification algorithm and attribute selection.

The authors of [6] in their investigation concluded that several different feature selection methods have been used in optimizing the N-gram system sequence features that classifies benign and malicious mobile applications. The authors used system call in Android as a feature in the experiment. In order to calculate the large sum of system call and processes for the detection approaches, they used the N-gram method. The combination of Wrapper as attribute selection and Linear SVM as classification algorithm improved the accuracy in classifying benign or malware applications.

The authors of [7] used several attribute selections with five algorithms in order to obtain a higher performance in malware classification. The objective of this study was to develop a mobile malware detection that provided a solution for protecting users from any malicious threat. In achieving this objective, the authors demonstrated the use of System Call Log as a feature in the experiment.

Similar works that used Android Permission systems were conducted in experiments by several authors. In [8], the author mentioned the problem of the growing number of users actually attracted hackers to develop malware applications to steal the private information and cause potential financial loses. The author analyzed an Android feature known as Permission to differentiate between the malware apps and goodware apps. However, to increase precision in result, the author used information gain as attribute selection and compared the three data mining algorithm to obtain better results.

A recent work [9] had proposed using Fest to build precise classification models by manually choosing features, and few of them used any feature selection algorithms to help pick typical features. The author used four features which are Permission, API, Action and IP and URL in order to prove the weakness of feature selection algorithm.

In [10], the authors used 3784 android applications to enhance the accuracy of malware classification based on Permission in APK file. The author used two methods to look for accuracy of classification. The feature selection was Gain Ratio Attribute Evaluator, ReliefF Attribute Evaluator, Cfs Subset Evaluator and Consistency Subset Evaluator and data mining algorithm, J48 Decision Tree, Bayesian, Classification, Random Forest, Classification and Regression Tree and SMO. Different from previous works, the authors of [11] applied the Bayesian algorithm in malware detection. The authors built a model to improve detection due to the emergence of Android malware sophistication in evading detection through the traditional signature-based scanners. API call was used to monitor suspicious activities that are running in the application.

In contrast to these works, this paper presents a pattern of data analysis in malware classification based on the comparison of all types of classification. Comparing all applications to the benign and malware is done in seven categories of classifiers (Bayes, Function, Lazy, Meta, MISC, Rules and Tree). Then, attribute selection was used as filtering to reduce the number of variables in Permission feature which involved eight attribute selection methods.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1

Paper	Platform	Feature Selection	Feature	Classifier	Result
[6]	Desktop	Wrapper	Spam mail, PDF	SVM	To validate that Feature Selection suit with classifier
[7]	Desktop	Info Gain	API	Random Forest	To detect malware and classifying it into appropriate malware families.
[8]	Android		APK file	K- Nearest Neighbor	To identify which machine learning are accurate in malware detection.
[9]	Android	Chi-Square & Relief F	APK File	Random Forest	To verify that both of feature selection given the best result in comparison to the individual algorithms results
[10]	Window OS	Principal Component	WinAPI Call	K-Nearest Neighbor	To reduce time in malware classification
[11]	Big Data	Chi-Square	API	Random Forest, Neural Network, SVM	To verify that combination of three machine learning algorithm produce an increasing accuracy in malware classification.
[12]		MIE	API Call	All Nearest Neighbor (ANN)	Using data mining to identifying new malicious executables.
[13]	Desktop	Mutual Information	API Call	Neural Network	To verify the effective of Neural Network in binary classification in order to classified malware family.

Finally, the attribute is limited to Permission feature that is included in protection level. xxx

3. METHODOLOGY

In this section, we present the methodology of our study; shown in Figure 1. This methodology involves four major processes. The first is APK file extraction which is the process to extract source code of files. The second process is Feature Vector Mapping; the process of mapping feature into binary. The third process is Attribute Selection that reduces the number of variables in feature. Lastly, the process of Classification aims to identify which file is benign or malware. The overall goal of this study is to determine the algorithm and attribute selection that are more efficient and effective in malware detection.

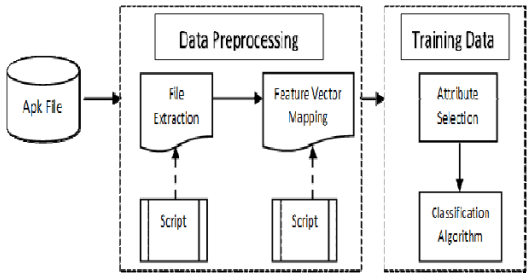

Figure 1: MalDroid in Architecture

3.1 Data Set

The android dataset used in this analysis were divided into two categories, benign and malware. The benign dataset was collected from Google Store while malware dataset was collected from several sources such as ContagioDump and AndroZoo. In order to ensure that the application is clean or contaminated by malware, the applications were uploaded into VirusTotal website to check their status. VirusTotal is an online scanner and it aggregates with many antivirus products (Sophos, Symantec, F-Secure, TrendMicro and MacAfee) to check for viruses that the user's own antivirus may have missed, or to verify against any false positives.

In this study, we used 8006 android applications, 3011 of which were found to be benign applications and 4995 malware applications. Each application was labelled as 'benign' or 'malware' to facilitate the process of supervised learning in attribute selection and classification.

3.2 Feature Extraction

APK stands for Android Package Kit, it was design for Android mobile operating system. APK file contains bytecode, configuration, precompiled library and resources files. Each APK file has its own AndoridManifest.xml. This file is the root to source code. This file contains all required files that are related to execute the application including Permission.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

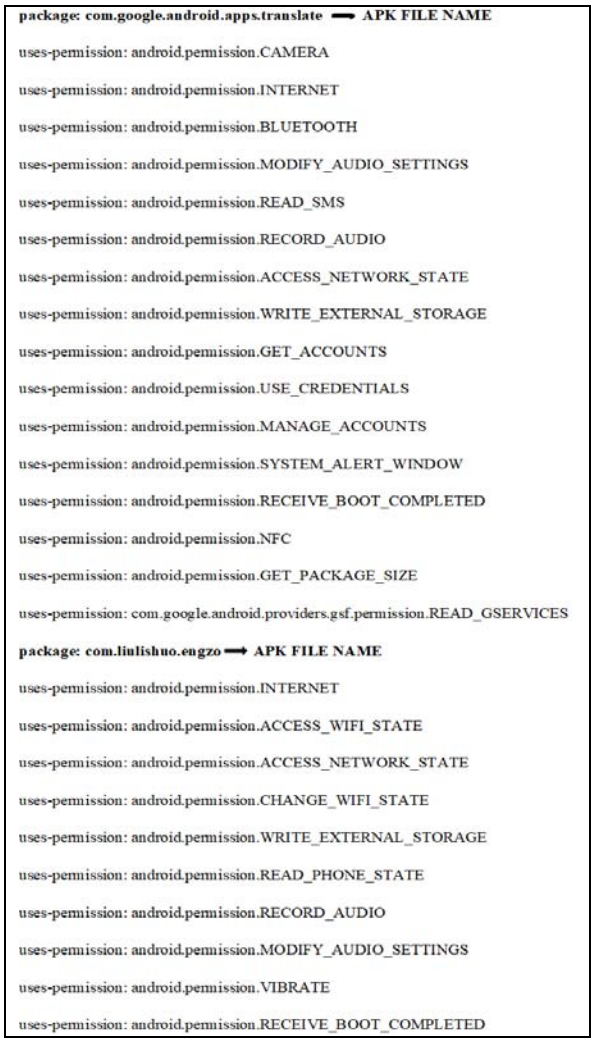

Figure 2: The Example of Apk File

Permission is part of Android Security and will act as a mechanism which restricts access of third-party Android applications to critical resources on the mobile system. The reason for a license is to ensure Android client's security. Android applications must be permitted to get to delicate client information, (for example, contacts and instant messages) just as certain framework highlights, (for example, camera and web). Contingent upon the element, the framework can consequently allow the authorization or incite the client to endorse the demand.

There are three processes involved to extract Permission feature. Firstly, we used Permission script in order to extract feature and to conduct our analysis. The advantage of using Permission script is that it allows for the running of a large number of APK file at one time. All the feature is represented in XLS format. Figure 2 below shows the example of two APK files that have been extracted (the package in bold shows the name of the APK file).

Then, each package will be converted into word format, saved and labelled using sequence number. Finally, we created a pattern data in feature set, and called it feature vector through editing the scripts in XLS format. Feature vector is the process of integrating all the Permissions that are required for this study. A discussion about the feature vector is detailed in the next section. This process is repeated to all applications and attributes are represented in binary format.

3.3 Features in Android File

Feature is the attributes used for defining the Permission characteristic of an application. Permission is an important component available in Android application. Without Permission, an application cannot run in android operating system because Permission allow access between the application and mobile device. Table 1 shows the example of several Permissions and their functions.

Table 1: Permission in Android File.

Permission	Function
uses-Permission: android.Permission .CAMERA	Request Permission to use a device's camera
uses-Permission : android.Permission .BLUETOOTH	Allows applications to connect to paired Bluetooth devices
uses-Permission: android.Permission.A CCESS_NETWORK_STATE	Allows applications to access information about networks
uses-Permission: android.Permission .INTERNET	Allows applications to open network sockets
uses-Permission: android.Permission. INSTALL_SHORTCUT	Allows an application to install a shortcut in Launcher

To assist users in understanding the importance of android security, Table 2 presents details about protection level in Permission. Permission protection has four levels which are [20] [21]:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Levels of Android Protection.

Protection Level	Description
Normal	This cover zones where your application needs to get to information or assets outside the application's sandbox, however where there's almost no hazard to the client's protection or the activity of different applications. For example, Permission to set the time zone is a normal Permission.
Dangerous	Control the sensitive data or resources that belong to user information, or could potentially affect the user's stored data or the operation of other apps. For example, READ_CONTACT allow and application to copy data from contact folder.
Signature	Only granted when application is signed with the device manufacturer certificate
Special	There are a few consents that don't carry on like typical and risky authorizations. SYSTEM_ALERT_WINDOW and WRITE_SETTINGS are especially touchy, so most applications ought not utilize them. In the event that an application needs one of these consents, it must pronounce the authorization in the show, and send an aim asking for the client's approval. The framework reacts to the purpose by appearing point by point the executives screen to the client.

In addition, starting from APLI level 23, Android version 6.0 Marshmallow, a new rule was applied where the application needs to have Permission every time it executes; this changed effected user awareness and caution at runtime.

Through feature extraction, we obtained hundreds of features. However, some of the features such as REQUEST_COMPANION_RUN_IN_BACKGROUND and READ_CALL_LOG were only used by a few applications. Moreover, some features such as SYSTEM_ALERT_WINDOW and WRITE_SETTINGS were found to belong to Special Permission and are widely used in benign and malware application that makes applications difficult to be classified as an actual malware or benign.

Therefore, we used features found in Normal and Dangerous Protection Level as our feature selection in order to reduce the complicacy of computation and low efficiency in building our malware detection model. Table 3 below shows the example of Attributes obtained from the process of Feature Extraction.

Table 3: List of Permission in Android.

Permission Name
1. ACCESS_LOCATION_EXTRA_COMMANDS
2. ACCESS_NETWORK_STATE
3. ACCESS_NOTIFICATION_POLICY
4. ACCESS_WIFI_STATE
5. ANSWER_PHONE_CALLS
6. ADD_VOICEMAIL
7. ACCESS_FINE_LOCATION
8. ACCESS_COARSE_LOCATION
9. BLUETOOTH
10. BLUETOOTH_ADMIN
11. BROADCAST_STICKY
12. BODY_SENSORS
13. CHANGE_NETWORK_STATE
14. CHANGE_WIFI_MULTICAST_STATE
15. CHANGE_WIFI_STATE
16. CAMERA
17. CALL_PHONE
18. DISABLE_KEYGUARD
19. EXPAND_STATUS_BAR
20. FOREGROUND_SERVICE
21. GET_PACKAGE_SIZE
22. GET_ACCOUNTS
23. INSTALL_SHORTCUT
24. INTERNET
25. KILL_BACKGROUND_PROCESSES
26. MANAGE_OWN_CALLS
27. MODIFY_AUDIO_SETTINGS
28. NFC
29. PROCESS_OUTGOING_CALLS
30. READ_SYNC_SETTINGS
31. READ_SYNC_STATS
32. RECEIVE_BOOT_COMPLETED
33. REORDER_TASKS
34.REQUEST_COMPANION_RUN_IN_BACKGROUND
35.REQUEST_COMPANION_USE_DATA_IN_BACKGROUND
36. REQUEST_DELETE_PACKAGES
37.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS
38. READ_CALENDAR
39. READ_CALL_LOG
40. READ_CONTACTS
41. RECORD_AUDIO
42. READ_PHONE_STATE
43. READ_PHONE_NUMBERS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

44. RECEIVE_SMS
45. READ_SMS
46. RECEIVE_WAP_PUSH
47. RECEIVE_MMS
48. READ_EXTERNAL_STORAGE
49. SET_ALARM
50. SET_WALLPAPER
51. SET_WALLPAPER_HINTS
52. SEND_SMS
53. TRANSMIT_IR
54. USE_FINGERPRINT
55. USE_SIP
56. VIBRATE
57. WAKE_LOCK
58. WRITE_SYNC_SETTINGS
59. WRITE_CALENDAR
60. WRITE_CALL_LOG
61. WRITE_CONTACTS
62. WRITE_EXTERNAL_STORAGE

3.3 Attribute Selection

Attribute selection is the process of selecting the prominent feature attributes from the dataset. Attribute selection method is utilized for identifying and removing irrelevant attributes from the data that does not give significance to the accuracy of a predictive model or decrease the experiment accuracy.

An attribute selection contains the combination of a search technique for recommending new feature subsets, along with an evaluation measure which scores the different feature subsets. The options of metric for evaluation is influenced greatly by the algorithm. The evaluation metric that distinguishes among the three feature selection algorithms category are wrappers, filters and embedded methods. Good and efficient attribute selection will contribute to effective performance gain through reducing the attributes and time spent in data analysis. In this study, we employed eight different attribute evaluator approaches with three search methods which are applied to find the best and effective attributor evaluator for machine learning algorithm.

Info Gain Attribute Evaluator is one method to evaluate the information gain. Info Gain evaluates the worth of an attribute by measuring the information gain with respect to the class. Class vary from 0 (benign) to 1 (malware). Those attributes that contribute more information will have a higher information gain value and can be selected, whereas those that do not add much information will have a lower score and can be removed. From the experiment, 44 from 62 features were ranked as having a high score based on the following formula:

Info Gain (Class, Attribute) = H(Class) – H(Class | Attribute), where H is the information entropy.

Correlation was measured through Pearson's correlation coefficient. The calculation of the correlation between each attribute and the output variable were done and only attributes that have a moderate-to-high positive or negative correlation (close to -1 or 1) were selected, while attributes with a low correlation (value close to zero) were dropped. Correlation-based filter gives high scores to subsets that include attributes that are highly correlated to the class attribute but have low correlation to each other. Let S is an attribute subset that has k attributes, rcf models the correlation of the attributes to the class attribute, rff the inter correlation between attributes.

$$MeritS = k\ rcf / sqrt(k+k(k-1)\ rff)$$

Using this formula, 6 attributes were found to be satisfactory for all attribute sets.

Gain Ratio (GR) quality assessment tackles the downside of Information Gain. Although Information Gain is normally a decent measure for choosing the pertinence of a characteristic, yet it is less productive when connected to properties that have diverse qualities. Gain Ratio intends to discriminate the multiplication of hubs and to be critical when information is uniformly spread and little when all information have a place within one branch. Normalization was applied to 44 attributes for information gain to contribute for classification process as below:

$$SplitInfoA\ (D) = -\sum vj=1\ (|Dj|/|D|)\ log2(|Dj|/|D|)$$

One R trait assessment is the strategy that assesses the value of quality by utilizing the OneR classifier. OneR, which stands for "One Rule", is a basic, yet exact, arrangement calculation that produces one guideline for every indicator in the information, at which point chooses the standard with the littlest absolute mistake as its "one principle". To make a standard for an indicator, we developed a recurrence table for every indicator to the objective. It was found that OneR produced leads that were somewhat less precise than best in class order calculations while producing results that are straightforward for people to translate.

Principal Component was also used in conjunction with ranker search. Dimensionality reduction was accomplished by choosing enough eigenvector to account for some of the variance in the original data –default 0.95%. Attribute noise

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

was filtered by transforming to the PC space, eliminating some worse eigenvector and then transformed back to the original space. The experiment showed that 44 out of 62 features were transformed into a new frame of reference that were specified by the most commonly used variance.

Attribute ranking was based on the most used attribute variance, followed by the second greatest variance, and so forth. Relief Attribute Evaluator is a technique that assesses the value of a property by over and again inspecting a case and considering the estimation of the given trait for the closest occasion of the equivalent and distinctive class. Alleviation ascertains an element score for each component which would then be able to be connected to rank and choose top scoring highlights for highlight determination.

Then again, these scores might be connected as highlight loads to control downstream displaying. Help highlight scoring depends on the recognizable proof of highlight esteem contrasts between closest neighbor example sets. On the off chance that a component esteem contrast is seen in a neighboring occasion combine with a similar class (a 'hit'), the element score diminishes.

SymmeticalUncertAttr is the method that evaluates the worth of an attribute by measuring the symmetrical uncertainty with respect to the class. It compensates for information gain's bias toward features with more values and normalizes its values to the range [0, 1] with the value 1 indicating that knowledge of the value of either one completely predicts the value of the other and the value 0 indicating that X and Y are independent [22]. It is based on the following equation:

$$SU=2/(Gain(A) / Info(D) + SplitInfo(A \quad))$$

The calculation results showed that 45 out of 62 attributes were normalized to the value range of 0 and 1.

3.4 Machine Learning in Classification

Machine learning provides the technical basis of data mining. It has been used to extract the raw data in dataset to information. One of the learning styles in machine learning is Classification [14]. Classification in malware analysis is the process to classify the data as either benign or malware. In this study we used classification algorithm from seven categories classification in machine learning to find which algorithm is more efficient and effective in malware classification. There are Bayes, Function, Lazy, Meta, Misc, Rules and Tree [15] [16] [17] [18]. The seven categories of machine learning are concluding as below:

Table 4: Levels of Android Protection.

Categories	Description	Example
Bayes	algorithms that use Bayes Theorem	Naïve
Function:	a function is being estimated	Linear Regression
Lazy:	lazy learning	k -Nearest Neighbor
Meta	Bunch of algorithms is combined or applied	Ensembles
Misc	execution that does not smoothly fit into different group categories	
Rules:	use rules	One Rule
Trees	use decision trees	Random Forest

4. RESULTS

The experiment in this study utilized Waikato Environment for Knowledge Analysis (Weka) version 3.8.2 and it was conducted on the Window platform with Intel processor i5 and 4Gb RAM. Total dataset used in this experiment are 8006 android applications; 40% from benign applications and 60% malware applications.

WEKA is a machine learning workbench that provide a comprehensive collection of machine learning algorithms and data preprocessing tools to researchers and practitioners. This tool has function such as classification, clustering and attribute selection. The WEKA system able to work with any kind of data from various field [28][29][30].

For the training and testing dataset, we engaged with k-fold cross validation to improve its validity. The process of cross validation involved three steps. Firstly, the dataset is divided randomly in k fold of equal size. Then, train the model on k-1 folds. Only one-fold was used for testing. Lastly, repeat this process of k until all the folds are used for testing.

The development of classifier algorithm analysis was performed in the following measurements of True Positive Rate (TP), False Positive Rate (FP) and Precision. This evaluation is consequential to four basic measures as below:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

True Positive	specific the number correctly identify benign application. The formula of True Positive is *TPR = TP/ (TP + FN)*
False Negative	specific the number incorrectly identify malware application. The formula of False Negative is *FPR = FP/ (TN + FP)*
True Negative	specific the number of identify malware application.
True Positive	specific the number of identify benign application.
Precision	The ration of retrieve instance that are relevant. The formula of Precision is *Precision = TP / (TP + FP)*

As presented in the previous section, the number of specific attributes is 62 based on the latest Android API level. The eight attributes selection and two search methods were used to determine the attribute ranking. Attribute ranking is an approach to structure an otherwise unorganized collection of computing data where the rank for

each data is based on the value of one or more of its attributes.

Attribute selection and search method applied in this study have produced many different prediction outputs depending on the classification algorithm used. The results of the analysis are shown in figure 2. Based on the analysis, we found that the attribute selection InfoGainEval and Ranker as search method using IBK algorithm are the best classification method in malware detection. The IBK algorithm returned the best result in TPR, FPR and precision along with increased speed of classification. The other algorithms also showed some significant results using the InfoGainEval. For instance, RandomCommitte algorithm is the second best with 0.894 TP rate followed by PART with 0.871 TP rate and BayesNet with 0.793 TP rate value.

In attribute selection approach, we have shown that IBK algorithm is a promising approach for InfoGainEval attribute selection. It outperforms most existing algorithms in terms of the number of selected features and classification accuracy. IBK based feature selection runs very efficiently on large datasets, which makes it very attractive for attribute selection in high dimensional data.

Attribute Evaluator	Search Method	Classifier	Training & Testing		
			TPR	FPR	Precision
Info Gain Evaluation	Ranker	BayesNeT	0.793	0.227	0.03s
		IBK	0.896	0.103	0s
		RandomComitte	0.894	0.105	1.27s
		PART	0.871	0.139	10.1s
CFS Sub Set Evaluation	Be First	BayesNeT	0.774	0.285	0.09s
		MultilayerPerceptron	0.786	0.257	9.42s
		IBK	0.787	0.253	0s
		RandomComitte	0.787	0.253	0.28s
		PART	0.786	0.254	0.23s
		RandomForest	0.786	0.254	1.53s
Correlation Atrr	Ranker	BayesNeT	0.793	0.227	0.31s
		IBK	0.793	0.227	0.01s
		RandomComitte	0.898	0.099	1.55s
		PART	0.875	0.131	13.18s
Gain Ratio Atrr	Ranker	BayesNeT	0.793	0.227	0.3s
		MultilayerPerceptron	0.876	0.133	170.04s

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

		IBK	0.896	0.103	0.02s
		RandomComitte	0.896	0.102	1.39s
		PART	0.871	0.139	10.91s
		RandomForest	0.9	0.101	7.99s
OneR Attri Eval	Ranker	BayesNeT	0.793	0.227	0.24s
		MultilayerPerceptron	0.885	0.113	297.72s
		IBK	0.898	0.102	0.01s
		RandomComitte	0.898	0.1	1.57s
		PART	0.875	0.131	14.71s
		RandomForest	0.901	0.098	9.22s
Principal Component	Ranker	BayesNeT	0.781	0.239	0.22s
		MultilayerPerceptron	0.863	0.147	169.85s
		IBK	0.872	0.138	0s
		RandomComitte	0.871	0.136	1.45s
		PART	0.851	0.166	8.21s
		RandomForest	0.874	0.137	8.88s
Relief Attribute Eval	Ranker	BayesNeT	0.793	0.227	0.28s
		MultilayerPerceptron	0.885	0.117	219.76s
		IBK	0.898	0.102	0.02s
		RandomComitte	0.9	0.096	1.64s
		PART	0.875	0.131	13.96
		RandomForest	0.901	0.099	8.58s
SymmeticalxUncert Attr	Ranker	BayesNeT	0.793	0.227	0.3s
		MultilayerPerceptron	0.881	0.121	159.75s
		IBK	0.896	0.103	0s
		RandomComitte	0.894	0.105	1.44s
		PART	0.871	0.139	12.4s
		RandomForest	0.898	0.103	8.86s

5. CONCLUSION

In this paper, we presented the analysis of malware detection that is concentrated through classification. Moreover, the motivation of this paper is to focus on identifying effective and efficient attribute selection and classification algorithm. In achieving this motivation which also was mentioned in Section 1, we used Weka as a tool in our experiment to compare each attribute and classification algorithm. The experiment showed that InfoGainEval is the most effective in attribute selection, while KNN algorithm is the best classification algorithm in malware detection. More specifically, the results showed that the combination of these two methods yielded a near 90% accuracy in identifying malware applications.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ACKNOWLEDGMENTS

This research paper and work is supported by University of Malaysia Pahang research grant, PGRS 170328. The author would like to thank you to Faculty of Computer Systems & Software Engineering for the facilities in supporting this research. Also, the author would to thank for En Azmi Amirnordin from University Teknologi Mara (Uitm) for the dataset of malware.

REFRENCES:

[1] "Android Security 2017 Year in Review: March 2018," no. March, 2018.

[2] G. Play, O. Windows, M. Office, W. Server, M. Block, and G. Play, "SophosLabs 2018 Malware Forecast," 2018.

[3] W. Wang, Y. Li, X. Wang, J. Liu, and X. Zhang, "Detecting android malicious apps and categorizing benign apps with ensemble of classifiers," Futur. Gener. Comput. Syst., 2017.

[4] A. Feizollah, N. B. Anuar, R. Salleh, and A. W. A. Wahab, "A review on feature selection in mobile malware detection," Digit. Investig., vol. 13, pp. 22–37, 2015.

[5] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "'Andromaly': A behavioral malware detection framework for android devices," J. Intell. Inf. Syst., vol. 38, no. 1, pp. 161–190, 2012.

[6] M. Zaki, S. Shahib, M. F. Abdollah, S. R. Selamat, and C. Y. Huoy, "A Comparative Study on Feature Selection Method for N-gram Mobile Malware Detection," Int. J. Netw. Secur., vol. 19, no. 5, pp. 1–7, 2017.

[7] M. Z. Mas'ud, S. Sahib, M. F. Abdollah, S. R. Selamat, and R. Yusof, "Analysis of Features Selection and Machine Learning Classifier in Android Malware Detection," 2014 Int. Conf. Inf. Sci. Appl., pp. 1–5, 2014.

[8] A. Altaher, "Classification of Android Malware Applications using Feature Selection and Classification Algorithms," VAWKUM Trans. Comput. Sci., vol. 10, no. 1, p. 1, 2016.

[9] K. Zhao, D. Zhang, X. Su, and W. Li, "Fest: A feature extraction and selection tool for Android malware detection," Proc. - IEEE Symp. Comput. Commun., vol. 2016–Febru, pp. 714–720, 2016.

[10] U. Pehlivan, N. Baltaci, C. Acarturk, and N. Baykal, "The analysis of feature selection methods and classification algorithms in permission based Android malware detection," 2014 IEEE Symp. Comput. Intell. Cyber Secur., no. December, pp. 1–8, 2014.

[11] S. Y. Yerima, S. Sezer, G. McWilliams, and I. Muttik, "A New Android Malware Detection Approach Using Bayesian Classification," 2013 IEEE 27th Int. Conf. Adv. Inf. Netw. Appl., pp. 121–128, 2013.

[12] F. Zhang, S. Member, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Evasion Attacks," pp. 1–12, 2015.

[13] S. S. Hansen, T. Mark, T. Larsen, M. Stevanovic, and J. M. Pedersen, "An Approach for Detection and Family Classification of Malware Based on Behavioral Analysis," 2016.

[14] F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani, "Evaluation of machine learning classifiers for mobile malware detection," Soft Comput., vol. 20, no. 1, pp. 343–357, 2016.

[15] L. D. Coronado-de-alba, A. Rodríguez-mota, and P. J. E.- Ambrosio, "Feature Selection and Ensemble of Classifiers for Android Malware Detection," pp. 0–5, 2016.

[16] O. Of, "Feature Selection and Extraction for Malware Classification," vol. 992, pp. 965–992, 2015.

[17] S. Computing and S. Computing, "2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)," pp. 560–566, 2016.

[18] Y. Fan, Y. Ye, and L. Chen, "Malicious sequential pattern mining for automatic malware detection," vol. 52, pp. 16–25, 2016.

[19] W. Huang and J. W. S. B, "MtNet : A Multi-Task Neural Network for Dynamic Malware Classification," vol. 2, pp. 399–418, 2016.

[20] "Permissions overview." [Online]. Available: https://developer.android.com/guide/topics/permissions/overview.

[21] J. Lendu, U. Teknikal, and H. T. Jaya, "ANDROID MALWARE CLASSIFICATION BASE ON APPLICATION CATEGORY USING STATIC CODE," vol. 96, no. 20, 2018.

[22] L. Yu and H. Liu, "Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution," Proc. Twent. Int. Conf. Mach. Learn., pp. 856–863, 2003.

[23] K. Chumachenko and I. Technology, "MACHINE LEARNING METHODS FOR MALWARE DETECTION AND," 2017.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[24] P. Goodwin, D. Onkal, and H. O. Stekler, "PT US CR," 2017.

[25] V. J. Saglani and A. V Gupta, "Using Machine Learning Algorithms," no. Icoei, pp. 1007–1012, 2018.

[26] M. Kedziora, P. Gawin, and M. Szczepanik, "M ALWARE D ETECTION U SING M ACHINE L EARNING A LGORITHMS A ND R EVERSE E NGINEERING O F A NDROID J AVA C ODE," vol. 11, no. 1, pp. 1–14, 2019.

[27] B. Biggio and F. Roli, "PT," Pattern Recognit., 2018.

[28] M. Hall et al., "The WEKA Data Mining Software: An Update," vol. 11, no. 1, pp. 10–18.

[29] Romli, R. N., Zolkipli, M. F., Al-Ma'arif, A., Ramli, M. R., & Salamat, M. A. (2018). Understanding the Root of Attack in Android Malware. International Journal of Integrated Engineering, 10(6).

[30] Romli, R. N., Zolkipli, M. F., Ali, A. M., & Ramli, M. R. (2018). Android: S-Based Technique in Mobile Malware Detection. Advanced Science Letters, 24(10), 7442-7445.