International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 3 Issue 6, October 2019 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Adaptive Mobile Malware Detection Model Based on CBR

Kyaw Soe Moe, Mya Mya Thwe

Lecturer, University of Computer Studies, Hpa-An, Myanmar

ABSTRACT

Today, the mobile phones can maintain lots of sensitive information. With the increasing capabilities of such phones, more and more malicious software (malware) targeting these devices have emerged. However there are many mobile malware detection techniques, they used specified classifiers on selected features to get their best accuracy. Thus, an adaptive malware detection approach is required to effectively detect the concept drift of mobile malware and maintain the accuracy. An adaptive malware detection approach is proposed based on case-based reasoning technique in this paper to handle the concept drift issue in mobile malware detection. To demonstrate the design decision of our approach, several experiments are conducted. Large features set with 1,065 features from 10 different categories are used in evaluation. The evaluation includes both accuracy and efficiency of the model. The experimental results prove that our approach achieves acceptable performance and accuracy for the malware detection.

KEYWORDS: *mobile security; machine learning; concept drift; cyber security; data mining*

How to cite this paper: Kyaw Soe Moe | Mya Mya Thwe "Adaptive Mobile Malware Detection Model Based on CBR" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-3 | Issue-6, October 2019, pp.231-238, URL: https://www.ijtsrd.com/papers/ijtsrd28088.pdf

IJTSRD28088

I. INTRODUCTION

The context of Industry 4.0 has led to the tremendous growth in mobile and smart device usage and mobile communication is becoming more and more important [1]. The mobile phones become the tools for productivity. The mobile phones also become communication devices that include several built-in sensors. When the application requires a sensor device or communication device, the mobile phones are great. The mobile technology can provide many conveniences to customers, allowing unlimited and ease of communication. As currently provided activities like electronic commerce, personal payments, social communication, entertainment activities, social gatherings, playing games and watching videos are feasible anywhere and anytime with mobile devices and the industrial can also be controlled with mobile devices. Unfortunately, the escalating number of users also provides hackers with the opportunity to develop various malicious applications and the topmost security concern for mobile services becomes the mobile malware which can violence all confidential information of the mobile user [2].

Mobile attacks are increasing and evolving from a variety of newer methods despite the use of a number of detection approaches to battle mobile attacks. According to Symantec report, the malware families increased around fifty eight percent between 2011 and 2013 and most of them were mobile malware [3]. Wombat Security revealed that 83% of organizations experienced mobile attacks such as mobile phishing in 2018 [4]. Figures published by the UK cyber security firm Alert Logic cited that phishing attacks, ransomware, and data loss as the top concerns [5]. Most of the mobile malwares frequently attacked the Android because the Android is an open-source operating system [6]. However the Android market supports many applications for users, there are many unofficial markets such as SlideME [7]. Moreover, the official Android market also cannot be able to closely control its application contents, for instance, 50 Android market applications were infected by Droid Dream malware in 2011[8]. Similarly, the mobile malware had been infected 35 applications in Google Play at least 10 months and these applications are downloaded around 9 million times [9]. Moreover, the worst cybercrimes such as advanced persistent threats (APTs) and ransom ware usually start from mobile malware [10].

Malicious apps utilize multiple methods to evade the existing detection mechanisms provided by Android operating system or existing anti-virus soft-ware. These evasion methods include dynamic execution, code obfuscation, repackaging or encryption. Sophisticated malware developers implement powerful encryption or obfuscation techniques to hide their malicious payloads from a detection system. That phenomenon is called concept drift in the context of machine learning and it becomes one of the most challenging issues for mobile malware detections. In most of the existing malware detection solutions, various individual machine learning algorithms were applied and showed some acceptable detection accuracy. These systems used specific feature patterns with selected algorithms [11], [12]. Unfortunately, most existing classification based malware detection

techniques could not afford to adapt automatically on the variation of input feature patterns [13]. Thus, an adaptive malware detection approach is required to be applied in malware detection models to prevent the degradation of detection accuracy in facing with concept drift.

In this work, a new way that can automatically adapt the classifiers based on the variation of input features pattern to improve the key quality criteria of malware detection, accuracy, and efficiency. The aim is to create a mobile malware detection model using a case-based reasoning approach for an automatic adaptation of classifiers according to the incoming feature patterns. By addressing the optimal selection of the suitable classifier to the incoming features using a case-based reasoning approach, the proposed mobile malware detection model could provide the best performance by combining the good performance of all used methods appropriately. An adaptive mobile malware detection system based on a case-based reasoning (CBR) technique, which can handle the concept drift challenge in malware detection, is proposed in this work. CBR is applied to construct a malware detection model. A knowledge base or case base will control the detection algorithm by utilizing the malware features as cases. Moreover, an experimental analysis to verify that our proposed case-based malware detection is suitable for handling concept drift of mobile attacks than existing detection approaches will be conducted.

The rest of the paper is organized as follows. In Section 2, some related works are discussed. The knowledge and theories required for our approach is presented in section 3. The section 4 presents the architecture of our model and the proposed model is evaluated in section 5. Finally, we conclude or work in section 6.

II. RELATED WORKS

Machine learning based malware detection system model the malware behaviour with some machine learning classifier and then used the model to detect the new malware. The machine learning (ML) classifier used dataset for input and constructs a model that is applicable to new data to identify pattern similarities. There are many studies with significant detection results [14], [15] by using such approach. The author of [16] evaluated four classifiers (i.e. AdaBoost, NB, DT48 and SVM) on application permissions for malware detection and achieved 81% accuracy using the Naïve Bayes classifier. Unfortunately, their approach is less effective for the malware which hides an updated version within the original application such as Basebridge.

Similarly, the best classification method is identified out of six classifiers, including DT, J48, NB, BN, k-Means, histogram and logistic regression, using the Andromaly framework in [17]. The feature selection methods such as Chi-square, Fisher score and information gain are adopted to enhance the detection accuracy in that framework. As a result, the J48 decision tree algorithm achieved 99.9% accuracy rate with information gain method. However they achieved the great accuracy, they used self-written malware to test their framework and cannot produce the realistic results.

A multi-level detector prototype is proposed by combining two system call levels: kernel and user level in [18]. The K-nearest neighbours (KNN) classifier is used with 12 system calls as the main features in that system and 93% accuracy rate for 10 malwares was successfully obtained. However those approach is promising, it cannot detect the malware that avoids the system call with root permission, for example SMS malware that is invisible in the kernel.

A malware detection system called RobotDroid is proposed based on the SVM classifier to detect the mobile malware in [19] . The focus was on privacy information leakage and hidden payment services. They evaluated three malware types, namely Gemini, DroidDream and Plankton. As a result, this framework is limited to few malware types and more would be required to increase detection accuracy.

A broad analysis of Android applications is performed to detect malwares in a system called DREBIN [20] and it collects permissions, hardware access, API calls, network address, etc. However, it is unable to detect malwares that use obfuscation technique and dynamically loaded code technique.

The aforementioned approaches demonstrate the rationale behind applying machine learning classifiers to detect mobile malware with specified features. However, these prior approaches cannot maintain the acceptable accuracy rate while it is handling the concept drift which can vary the input features. Thus, an adaptive malware detection approach is proposed with case-based reasoning (CBR) technique is this work.

III. THEORETICAL BACKGROUND

A. Case-Base Reasoning

Case-Base Reasoning (CBR) is a problem solving approach that solves new problems by adapting or re-using old solutions that were used to solve similar problems [21]. The past experience or previous problems are saved as cases and each case contains representative features, characteristics of the problem and its solution. The case-base is a collection of these cases. The knowledge base of the problem solving experience is used for the new problem solving [22]. The solutions in the retrieved cases are reused as a proposed solution to the new problem. Thus, the solution to the new problem can be found from similar known solution in the past.

If the new problem situation is exactly as same as the previous cases, then the reuse is simple. CBR systems start their reasoning from the knowledge unit, called cases, while the data mining systems most often start from the raw data. CBR systems also belong to the instance based learning systems in the field of machine learning, that are defined as a system that is capable of automatically improving their performance over time. As long as the CBR systems learn new cases in the retain step, they are qualified as the learning systems, thus belonging to the machine learning system [23].

B. Types of Malware

In this section, give a brief introduction to common malware principles. Malware is a portmanteau of the two words malicious and software, which clearly indicates that malware, is a program with malicious intentions. In order

to understand what these malicious intentions actually are, we introduce the terms: infection vector and infection payload.

The infection vector describes which techniques are used to distribute the malicious application. Several known approaches are: e.g. file injection, file transport, exploit2, or boot sector corruption. The infection payload represents the actual content that is used to harm the victims' machine. Several known possibilities for payloads are deleting files, denying service, or logging keystrokes.

There are three common categories of malicious software: virus, worm, and Trojan horses. A virus mostly comes in a hosting medium. If the user executes this file, the virus processes its' malicious commands which can be almost everything the OS allows. A worm can often spread without user interaction. Once started, it searches for infect-able victims in range. If a victim is found, it normally uses an exploit to attach itself to the victim and then repeats this behavior. Sometimes worms drop other malware that can be back-doors that allow remote access. Bot programs installed this way can make the victim to remote triggered attacks. A Trojan horse is a program that is disguised to pursue a user to install it. However, it is not possible to categorize every malware clearly, Table 1 illustrate the general overview on the malware characteristics.

TABLE1: CHARACTERISTICS OF SOME MALWARES

Type	Appearance	User Interaction	Vector	Payload
Virus	Needs a hosting medium	Usually needed	Such as file injection or boot sector	Such as system modification
Worm	Independent program	Usually not needed	Such as exploit	Such as malware drop
Trojan Horse	Malicious functionalities disguised	Usually needed	Such as email attachment or download	Such as backdoors

IV. ARCHITECTURE OVERVIEW

An adaptive malware detection model is proposed with a case-based reasoning approach. The design for developing the cased-based adaptive classification system is proposed in this section. Two main parts including Android smartphones based input system and cloud environment based detection system. The overall architecture is illustrated in Figure 1.

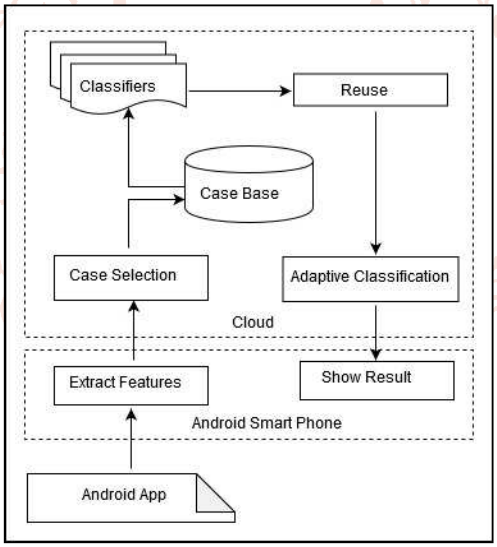

Figure 1. Overall architecture

The feature for malware detection will be extracted from the Android APP as illustrated in Figure 1. The information about the extracted feature will be discussed in next section. As the cloud environment will be used for main detection process, the extracted features will be sent to the cloud. The main goal of this work is to enhance the malware detection process and our detection processes will used both static and dynamic feature from Android malware dataset which are described in Table 2. The detailed process of feature extraction is out of this paper scope.

TABLE 2: FEATURE SETS

No	Features sets	Features count	Example features
1	Android components	76	android.media, android.media.effect, android.media.audiofx, android.service.textservice, android.service.notification
2	API counts	31	account_information, account_settings, audio, bluetooth, bluetooth_information
3	API usage actions	82	android.util, android.widget, android.renderscript, android.webkit, android.os, android.os.storage, android.content
4	Security sensitive flows	421	system_settings___audio, system_settings_phone_connection, system_settings_voip, system_settings_database_information

5	Hardware components	6	android.hardware.display, android.hardware, android.hardware.usb, android.hardware.location, android.hardware.input
6	Intent_action	109	action_main, action_view, action_default, action_attach_data, action_edit, action_insert_or_edit
7	Permission	82	android.permission.access_cache_filesystem, android.permission.access_checkin_properties, android.permission.access_coarse_location, android.permission.access_gps
8	shell_command_strings	190	runtime.exec, createSubprocess, Cipher-classes, longstring, SecretKey, method.invoke, small_code_size
9	contentvisual	19	HostnameLength, PathLength, QueryLength, DoubleSlashInPath, NumSensitiveWords, EmbeddedBrandName, PctExtHyperlinks,
10	URLs	49	having_ip_address, url_length, shortining_service, having_at_symbol, double_slash_redirecting, prefix_suffix,
	Total	1,065	

The features are extracted from more than 10,000 Android malware samples. The malware samples are collected from Android malware repositories including VirusShare [24], AndroZoo [25], Droid screening [26] and Reveal droid [27]. The features include 31 features of API counts, 82 features of API usage actions, 421 features of security sensitive flows, 6 features of hardware components, 109 features of intents, 82 features of permissions, 190 features of malicious shell command and strings, 19 features of content visual and 49 features of URLs. Thus, there are 1,065 features in total.

Our main contribution start with the receiving of the extracted malware features. The first most process is the retrieving the most similar case from the case-base that is a storage of previous Android malware detection along with the corresponding features. The case-base is set up before the case retrieving process. Figure 2 illustrates the case-base setting up process.

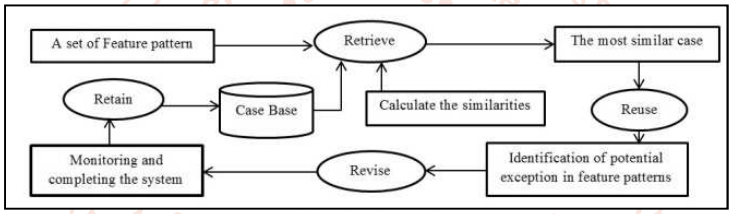

Figure 2. Setting up Case Base

According to the retrieved case, the most suitable classification techniques will be used for the adaptive classification. If the feature set extracted from the Android application does not match the sets of features stored in the case-base, the adaptive classification will select the suitable methods to process the extracted feature set according to the similarity ratio score. The selection of suitable methods means choosing the multiple classifiers for the extracted feature set. Finally, the final result of the active Android application will be sent to the application on Android smartphone to be displayed to the user.

A. Case Representation

A case represents an experience at an operational level. Typically, a case includes the problem specification, the solution and sometimes the outcome. This is the most common representation used. However, more elaborate case representations can be employed. Depending on the information included in a case, different types of results can be achieved from the system. Cases that describe a problem and its solution can be used to derive solutions to new problems.

In general, a case specification is described as a set of features. The features are those aspects of the domain and the problem that are considered to be most significant in determining the solution and/or outcome. A case represents an experience. In this situation, a case should represent the features of the application that is used to determine a malware attack.

In our model, a case includes the combination of feature sets, ensemble method of classifiers or individual classification algorithm with their specific parameters, the accuracy and performance of the solution, and potential facilitations. A case description stored in the malware detection system is shown in Table 3.

TABLE 3: A CASE DESCRIPTION FOR MOBILE MALWARE DETECTION SYSTEM

No	Name	Value
1	Case ID	Case identification Number
2	Feature pattern	Combination of feature sets
3	Ensemble methods of classifiers (or) Classification Algorithm	Boosting / Bagging / Bayesian (or) Algorithm name and their specific parameters
4	Accuracy	Percentage of correctly classification
5	Performance	Runtime (seconds)

To define a new case in case-base, the input features have to pass through different machine learning classifiers, and the results from each classifiers are calculated to produce the final result. Then, the input features, the classifiers with parameters, the activation function, and the final result are stored in the case-base (knowledge base) as a new case. The process of defining a new case to be stored in the case-base is shown in Figure 3.

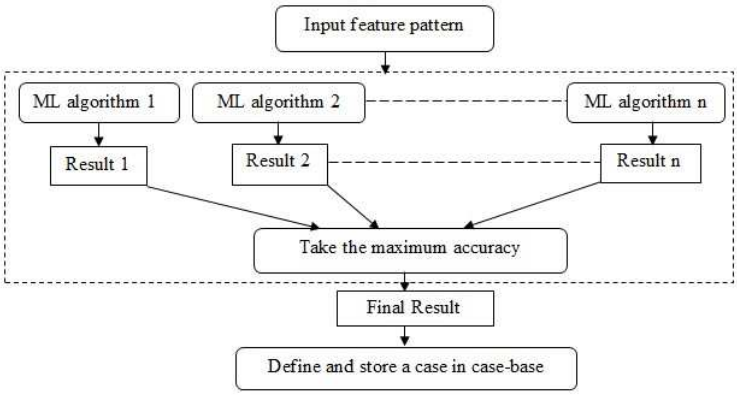

Figure 3. Defining a New Case

B. Case Retrieval

Case-Based Reasoning (CBR) solves a new problem by retrieving the previously solved problems and their solutions from a knowledge source of cases, called the case-base. There are challenges related to retrieving process that still need to be addressed. One issue is the computation of similarity, which is particularly important during the retrieving process. The effectiveness of a similarity measurement is determined by the usefulness of a retrieved case in solving a new problem.

The aim of using the CBR approach is the selection of the most similar past malware detection cases to the new problem. A set of similar cases is selected from the case-base according to a similarity criterion that requires the specification of weights corresponding to attributes. The assessment of case similarity involves the comparison of attribute values of the new case and that of the past cases, stored in the case-base. The retrieved old cases are ranked according to their similarity scores to the attributes of the new case. In this work, the nearest neighbour method is applied to calculate the similarity score and the total similarity score of a potentially useful case.

C. Adaptive Classification System Design

The main objective of case-based adaptive classification is to assign a suitable classification technique to the target case (a feature set extracted from Android application) by identifying and analysing the training case (sets of features that stored in the case-base) that is similar. The proposed case-based adaptive classification is shown in Figure 4. If the feature set extracted from the active Android application do not match with any set of features, stored in the case-base (that means the extracted feature set is not complete for the case retrieving process), the adaptive classification will select suitable methods to process the extracted feature set. The selection of suitable methods has two options. First, the possible features are added to the extracted feature set in order to perform the case retrieving process and to choose a suitable classifier. Second, multiple classifiers are selected to process the extracted incomplete feature set. Under the second option, multiple answers, resulted from multiple classifiers are collected in order to produce a final answer by the way of weighted sum of all answers.

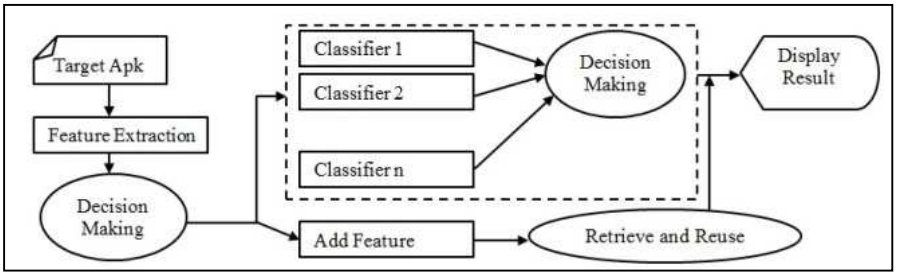

Figure 4. Adaptive Classification

V. DETECTION MODEL AND EVALUATION

This section explains how our detection model performs adaptively on the combination of individual classifiers and ensemble classifier. To verify that our proposed model can improve the accuracy of the mobile malware detection, an experiment is conducted using the feature sets which have been described in Section IV. The experiment was conducted by running Weka 3.8 on a Laptop computer with core i7 processor, 8 GB RAM, and Windows 8.1 64-bit operating system. The cross-validation method is used as an evaluation technique to estimate the error rate efficiently and in unbiased way by running repeated percentage splits. Firstly, the dataset is divided into 10 pieces. Each piece is used as a testing data set in turn while the remaining 9 pieces together are used as a training data set. We preformed 10 simulations (i.e. experiments are repeated 10 times). Then, all these results are averaged as a single estimation result. Six of the existing machine learning algorithms are chosen from different categories and used with 10-fold cross validation methods to evaluate the variation of accuracy and efficiency.

A. Machine Learning Classifiers

To detect and classify the malware applications, different machine learning classification techniques are used with an adaptive method. An adaptive classification system is proposed to automatically choose a combination of suitable classifiers for the extracted features of an active Android application. Various machine learning techniques were used as the classifier in existing works [28], [29], [30], [31]. Among them, six algorithms were selected from different categories for the coverage usage of all classification nature. The six algorithms include C4.5 (J48), decision table (DT), k-Nearest Neighbors (IBK), Logistic Regression (LR), Naive Bayes (NB), and support vector machine (SVM). According to the pretesting on the effectiveness of parameter on these classifiers, Naive Bayes (NB) classifier with supervised discretization function, the default maximum number of iterations in Logistic Regression (LR) , the confidence factor of 0.5 for pruning tree for J48 classifier, and a 1-Nearest Neighbors (IBK) classifier are chosen for our experiment. SVM and Decision table classifiers are used with their default parameters.

B. Experimental Results and Analysis

The accuracy comparison of six classifiers on the 10 feature sets is shown in Table 4. It can be seen that the accuracy of each classification algorithm depends on the features. IBK can provide a better accuracy in 6 features and J48 can provide a better accuracy in other 4 features. Our work aims to detect mobile malware in the nature of feature independent with various classifiers. To create a real-world application, a random feature combination is created because a new Android application can consist of any combination of features. In this experiment, 5 random combinations of features are created, as shown in Table 5.

TABLE 4: ACCURACY COMPARISON OF CLASSIFIERS ON 10 FEATURES

	Feature Sets	J48 (%)	DT (%)	IBK (%)	LR (%)	NB (%)	SVM (%)
1	android components	93.23	89.02	93.40	90.16	84.67	87.95
2	API count	95.85	93.02	95.66	91.90	89.20	85.25
3	APIusage_actions	95.20	91.86	95.32	91.97	89.02	91.24
4	flow	93.05	91.03	93.32	87.18	87.45	83.17
5	Hardware components	89.00	89.06	89.12	89.06	89.02	89.06
6	intent_action	86.89	85.73	87.13	84.64	83.75	85.53
7	permission	94.30	91.92	94.65	93.95	88.54	94.14
8	shell_command_strings	75.40	71.18	74.08	70.28	68.74	70.22
9	content_visual	97.20	95.79	95.53	94.49	95.77	93.87
10	URLs	96.03	93.24	97.18	93.99	92.98	93.80

TABLE 5: SCENARIOS FOR RANDOM COMBINATIONS OF FEATURES

Case ID	Feature Pattern	Combination of Feature Sets	Number of features
01	Pattern 1	API count + API usage + Hardware	112
02	Pattern 2	API count+ Intent	139
03	Pattern 3	API count + API usage + Intent + Hardware	220
04	Pattern 4	Flow + Intent	529
05	Pattern 5	Flow + Intent+ API usage+ Hardware	610

These 5 feature combination patterns are tested with individual six classifiers and three models of ensemble classifiers to develop a case for our adaptive model. Each model is an ensemble of six classifiers with different method in providing the final answer. The final answer finding methods of ensemble classifiers include the average of probabilities, majority voting, and maximum probabilities. The detection results for 5 scenarios of random feature combination sets with the six base classifiers and three ensemble classifiers are described in Table 6.

According to the results shown in Table 6, some feature patterns are more suitable with ensemble techniques while some are better used with individual classification techniques. It can conclude that the accuracy variation of classification techniques in mobile malware detection are heavily relying on the input features.

TABLE 6: DETECTION ACCURACY OF 5 SCENARIOS ON RANDOMLY COMBINED FEATURE PATTERNS

Case ID	J48 (%)	DT (%)	IBK (%)	LR (%)	NB (%)	SVM (%)	AVG (%)	MAJ (%)	MAX (%)
01	95.93	93.07	95.45	92.47	89.42	91.62	95.31	95.31	92.87
02	94.72	91.62	94.04	90.18	86.44	89.27	94.26	94.20	91.38
03	96.32	92.67	95.60	94.89	90.69	92.57	96.43	96.41	94.31
04	90.56	86.38	90.45	88.51	81.55	87.88	90.64	90.64	88.52
05	95.33	89.69	94.37	93.97	92.28	91.61	95.68	95.69	92.68

The adaptive method used in our model will choose the most suitable classification approach for a set of input features. Based on the results presented in Table 6, we can develop a case to be stored in case-base for an adaptive choice of suitable classifiers. The tentative cases for building our case-based malware detection model is shown in Table 7.

TABLE 7: ACCURACY AND EFFICIENCY OF PROPOSED ADAPTIVE MODEL

Case ID	Feature Pattern	Adaptive method	Accuracy (%)	Run time (seconds)
1	Pattern 1	J48	95.93	4.43
2	Pattern 2	J48	94.72	4.54
3	Pattern 3	AVG	96.43	95.18
4	Pattern 4	AVG, MAJ	90.64	174.4, & 174.6
5	Pattern 5	MAJ	95.69	205.50

In order to assess the effectiveness of our proposed model, the confusion matrix evaluation is applied: Accuracy, Precision and Sensitivity. While sensitivity expresses the ability of a model to find all relevant instances in the dataset; precision expresses the proportion of the instances that our model predicts as positive and they are actually positive. The following formulas represent their definitions:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

The evaluation of effectiveness on our proposed model by means of accuracy, precision and sensitivity is described in Table 8.

TABLE 8: DETECTION RESULTS ACHIEVED BY THE PROPOSED MODEL

Case	Classifier	Accuracy (%)	Precision (%)	Sensitivity (%)
01	J48	95.96	83	79
02	J48	94.66	87	86
03	AVG	96.45	92	75
04	AVG	90.77	84	62
05	AVG	95.80	90	74

According to the results shown in Table 8, our adaptive model achieves a good detection accuracy for the malware features. Meanwhile, the performance of all the classifiers gets an acceptable precision and sensitivity ratio. According to the previous experiments, our adaptive malware detection model using case-based reasoning can perform well on the diversely distributed features.

VI. Conclusion

In this paper, an adaptive mobile malware detection model based on a variation of input feature patterns using a case-based reasoning (CBR) technique is proposed. An experimental analysis is conducted to demonstrate the design decision of our model and to verify the performance of our proposed model in handling the concept drift of mobile malware attacks. The proposed model is evaluated with a large feature set that contains 1,065 features from 10 feature groups which are frequently collected from Android apps. Moreover, 5 cases of randomly combined patterns of features are created in order to provide a diversity of unknown patterns to mimic a new real-world mobile apps. Six classification algorithms are chosen from different categories for the coverage usage of all classification nature on the diversion of feature sets. Three ensembles of six base classifiers are used. Each of which uses different final answer finding methods including average, majority voting, and maximum. Total, there are 9 classifiers. By addressing the optimal selection of the suitable classifier to the incoming features using a case-based reasoning approach, the proposed mobile malware detection model could provide an accuracy improvement with an acceptable runtime increment.

References

[1] M. Schroeck, J. Kawamura, and A. Kwan, "Strategic guardrails for digital transformation | Deloitte Insights." [Online]. Available: https://www2.deloitte.com/us/en/insights/focus/industry-4-0/digital-transformation-strategic-guardrails.html. [Accessed: 02-Sep-2019].

[2] M. Vergelis and T. Shcherbakova, "Spam and phishing in Q2 2019." [Online]. Available: https://securelist.com/spam-and-phishing-in-q2-2019/92379/. [Accessed: 02-Sep-2019].

[3] "Android Ransomware Predictions Hold True," Symantec Security Response. [Online]. Available: http://www.symantec.com/connect/blogs/android-ransomware-predictions-hold-true. [Accessed: 02-Sep-2019].

[4] B. Sussman, "2019 State of the Phish Report." [Online]. Available:https://www.secureworldexpo.com/industry-news/state-of-the-phish-2019-report. [Accessed: 02-Sep-2019].

[5] L. Wu, X. Du, and J. Wu, "Effective Defense Schemes for Phishing Attacks on Mobile Computing

Platforms," IEEE Trans. Veh. Technol., vol. 65, no. 8, pp. 6678–6691, Aug. 2016.

[6] P. Teufl, M. Ferk, A. Fitzek, D. Hein, S. Kraxberger, and C. Orthacker, "Malware detection by applying knowledge discovery processes to application metadata on the Android Market (Google Play)," Secur. Commun. Netw., vol. 9, no. 5, pp. 389–419, 2016.

[7] "SlideME | Android Apps Market: Download Free & Paid Android Applications." [Online]. Available: http://slideme.org/. [Accessed: 02-Sep-2019].

[8] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for Android," presented at the Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, 2011, pp. 15–26.

[9] "More 'BadNews' for Android: New malicious apps found in Google Play | Ars Technica." [Online]. Available: https://arstechnica.com/information-technology/2013/04/more-badnews-for-android-new-malicious-apps-found-in-google-play/. [Accessed: 02-Sep-2019].

[10] M. Moghimi and A. Y. Varjani, "New rule-based phishing detection method," Expert Syst. Appl., vol. 53, pp. 231–242, Jul. 2016.

[11] S. Wang, Z. Chen, Q. Yan, B. Yang, L. Peng, and Z. Jia, "A mobile malware detection method using behavior features in network traffic," J. Netw. Comput. Appl., vol. 133, pp. 15–25, May 2019.

[12] S. Liang and X. Du, "Permission-combination-based scheme for Android mobile malware detection," in 2014 IEEE International Conference on Communications (ICC), 2014, pp. 2301–2306.

[13] "Android malware detection through hybrid features fusion and ensemble classifiers: The AndroPyTool framework and the OmniDroid dataset," Inf. Fusion, vol. 52, pp. 128–142, Dec. 2019.

[14] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical Real-time Intrusion Detection Using Machine Learning Approaches," Comput Commun, vol. 34, no. 18, pp. 2227–2235, Dec. 2011.

[15] T. Chen, Q. Mao, Y. Yang, M. Lv, and J. Zhu, "TinyDroid: A Lightweight and Efficient Model for Android Malware Detection and Classification," Mobile Information Systems, 2018. [Online]. Available: https://www.hindawi.com/journals/misy/2018/4157156/. [Accessed: 07-Sep-2019].

[16] C.-Y. Huang, Y.-T. Tsai, and C.-H. Hsu, "Performance Evaluation on Permission-Based Detection for Android Malware," in Advances in Intelligent Systems and Applications - Volume 2, 2013, pp. 111–120.

[17] 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.

[18] G. Dini, F. Martinelli, A. Saracino, and D. Sgandurra, "MADAM: A Multi-level Anomaly Detector for Android Malware," in Proceedings of the 6th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security: Computer Network Security, Berlin, Heidelberg, 2012, pp. 240–253.

[19] "RobotDroid: A Lightweight Malware Detection Framework on Smartphones - ProQuest." [Online]. Available: https://search.proquest.com/openview/0fc0b66946bbe519a62672c1b88a51dd/1?pq-origsite=gscholar&cbl=136095. [Accessed: 07-Sep-2019].

[20] "Drebin: Effective and Explainable Detection of Android Malware in Your Pocket – NDSS Symposium." .

[21] M. M. Richter and R. O. Weber, Case-based reasoning. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[22] S. Craw, N. Wiratunga, and R. C. Rowe, "Learning adaptation knowledge to improve case-based reasoning," Artif. Intell., vol. 170, no. 16, pp. 1175–1192, Nov. 2006.

[23] S. Begum, M. U. Ahmed, P. Funk, N. Xiong, and M. Folke, "Case-Based Reasoning Systems in the Health Sciences: A Survey of Recent Trends and Developments," IEEE Trans. Syst. Man Cybern. Part C Appl. Rev., vol. 41, no. 4, pp. 421–434, Jul. 2011.

[24] "VirusShare.com." [Online]. Available: https://virusshare.com/. [Accessed: 25-Jan-2019].

[25] K. Allix, T. F. Bissyandé, J. Klein, and Y. Le Traon, "Androzoo: Collecting millions of android apps for the research community," in Proceedings of the 13th International Conference on Mining Software Repositories, 2016, pp. 468–471.

[26] J. Yu, Q. Huang, and C. Yian, "DroidScreening: a practical framework for real-world Android malware analysis," Secur. Commun. Netw., vol. 9, no. 11, pp. 1435–1449.

[27] "joshuaga / RevealDroid." [Online]. Available: https://bitbucket.org/joshuaga/revealdroid. [Accessed: 28-May-2018].