Research Article

A Case-Based Reasoning Approach for Automatic Adaptation of Classifiers in Mobile Phishing Detection

San Kyaw Zaw and Sangsuree Vasupongayya

Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University, Hatyai, Songkhla 90112, Thailand

Correspondence should be addressed to Sangsuree Vasupongayya; vsangsur@coe.psu.ac.th

Received 5 April 2019; Accepted 28 May 2019; Published 19 June 2019

Guest Editor: Arash H. Lashkari

Copyright © 2019 San Kyaw Zaw and Sangsuree Vasupongayya. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Currently, the smartphone contains lots of sensitive information. The increasing number of smartphone usage makes it more interesting for phishers. Existing phishing detection techniques are performed on their specific features with selected classifiers to get their best accuracy. An effective phishing detection approach is required to adapt the concept drift of mobile phishing and prevent degradation in accuracy. In this work, an adaptive phishing detection approach based on case-based reasoning technique is proposed to handle the concept drift challenge in phishing apps. Several experiments are conducted in order to demonstrate the design decision of our proposed model. The proposed model is evaluated with a large feature set containing 1,065 features from 10 different categories. These features are extracted from more than 10,000 android applications. Five combinations of features are created in order to mimic new real-world Android apps to evaluate our experiments. Moreover, a reduced feature set is also studied in this work in order to improve the efficiency of the proposed model. Both accuracy and efficiency of the proposed model are evaluated. The experimental results show that our proposed model achieves acceptable accuracy and efficiency for the phishing detection.

1. Introduction

Mobile communication is becoming more and more important within the context of Industry 4.0 [1]. The topmost security concern for mobile services is the phishing attack which can violence all confidential information of the mobile user [2]. Phishing attacks are increasing and evolving from a variety of newer methods despite the use of a number of detection approaches to battle mobile phishing attacks. Wombat Security revealed that 83% of organizations experienced phishing attacks in 2018 [3]. Figures published by the UK cyber security firm Alert Logic cited that phishing attacks, ransomware, and data loss as the top concerns [4]. Moreover, cybercrimes such as advanced persistent threats (APTs) and ransomware often start from phishing [5]. Currently, the phishers certainly try to hide their malicious payloads from a detection system using methods such as emulator detections, applications icon hiding, and reflection. APWG Phishing Attack Trends Reports released in March 2019 said a detection of phishing sites has become harder because phishers were obfuscating phishing URLs with

multiple redirections [6]. In the context of machine learning, this phenomenon is known as concept drift and it becomes the main challenge to mobile phishing detections. Thus, the machine learning classifiers, applied in phishing detection models, must adapt to this concept drift in order to prevent any degradation in their detection accuracy.

In earlier phishing detection works, the variation of individual machine learning classification algorithm was applied. Each earlier phishing detection approaches showed an acceptable detection accuracy while using specific feature patterns with selected detection algorithms in their specific application domain [7, 8]. Currently, the usage of individual classification algorithm in phishing detection is developing to a combination of multiple classifiers in the form of ensemble methods to produce a better accuracy with more efficiency [9–11]. Unfortunately, most existing ensemble classification techniques in phishing detection could not afford to adapt automatically on the variation of input feature patterns, and it remains as a challenging issue in the phishing detection works [12]. Therefore, finding a way to make automatic adaptation classifiers based on the variation

of input features pattern will improve the key quality criteria of phishing detection, accuracy, and efficiency.

The main objective of this work is to create a mobile phishing detection system 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 phishing detection system could provide the best performance by combining the good performance of all used methods appropriately. An adaptive phishing detection system based on a case-based reasoning (CBR) technique, which can handle the concept drift challenge in phishing apps, is proposed in this work. CBR is applied to construct a phishing detection model. A knowledge base or case base will control the detection algorithm by utilizing phishing features as cases. Moreover, an experimental analysis to verify that our proposed case-based phishing detection is suitable for handling concept drift of mobile phishing attacks than existing detection approaches will be conducted.

The rest of the paper is organized as follows. In Section 2, the background information of phishing attacks on smartphone is presented. The machine learning techniques on phishing detection and the background of case-based reasoning which models the ensemble of classifier approaches as cases in a knowledge base are also illustrated. Next, the overview architecture of the proposed adaptive phishing detection system and their detail processes are described in Section 3. The accuracy and performance analysis of the proposed system is presented in Section 4. The conclusions are described in Section 5.

2. Theoretical Background

The background technologies are described in this section. The nature of phishing attacks on smartphones and their attack techniques are presented in Section 2.1. Section 2.2 presents the literature review on existing phishing detection solutions based on machine learning techniques and their frequently used features. Lastly, case-based reasoning classification techniques are explained in Section 2.3.

2.1. Phishing on Smartphone. Nowadays, phishers are motivated to target smartphones due to several different reasons. A smartphone today is as powerful as a desktop or laptop computer. Smartphones usually contain lots of sensitive information of their owners. The increasing number of smartphone usage makes it more interesting for attackers. The phishing attack techniques are based on two categories including application-oriented phishing attacks and website-oriented phishing attacks.

The application-oriented phishing attacks can be categorized into two types based on their launching methods. First, the phishing application attempts to hijack (task interception) existing legitimate applications and continuously performs task polling. The phishing application will launch itself as long as it detects the launch of the target applications. These task interception attacks are specially based on

the fake graphical user interface (GUI) techniques, which can easily impersonate and are hard to detect since a large touch screen is used as the primary user interface on most smartphones. As a result, the fake login interface is layered over the top of the real one, and the phishing app appears to be the target app. Second, the phishing application (repackaged applications) can directly present itself as the targeted legitimate app. This event may occur when the user downloads the fake applications from an unofficial app market. The website-oriented phishing attacks can also be categorized into two types based on their techniques. First, a phishing website hides (spoof) the URL bar of the targeted websites. Second, a phishing website attempts to overlay the genuine website with a crafted pop-up window. The spoofing URL is the process of creating a fake or a forged URL which impersonates a legitimate and secure website. This kind of URL spoofing attack is harmful and dangerous because the website looks exactly like the original one [13]. The fake website asks the user to enter his/her username, password, credit card number, or other information. For a legitimate mobile app that includes an embedded web page served over HTTP or a legitimate mobile app that allows the overlaying pop-up window, the network attacker can change the login button on the page or substitute a crafted pop-up window so that there is a link to a page owned by the attacker. When the user clicks the button, the user will be taken to the phishing page within the embedded web frame. This way, the attacker can steal the user credentials. The attacker can then relay the credentials to the valid website in order to mimic the normal work flow.

Existing solutions in phishing detection show an acceptable accuracy in their specific domain using their targeted features and their specified machine learning techniques. Thus, an effective phishing detection that is less dependent on the features pattern is still needed in this age. This work aims to propose an adaptive phishing detection by combining many existing techniques.

2.2. Current Phishing Detection Solutions. Existing phishing countermeasures use techniques such as content filtering, visual matching, and blacklist or whitelist matching [14]. Content filtering system examines the content of webpages for suspected URLs. Content filtering can be achieved by identifying statistical differences between legitimate and suspected phishing contents or constructing a set of rules [15, 16]. Visual matching computes a visual similarity between the phishing and the legitimate pages, based on the images, blocks, and layout [17, 18]. For a blacklisting system, the known phishing URLs are listed based on a human verification method. The very low false positive rates will be resulted in this approach. For a whitelisting system, users specify the links of trusted sites and frequently accessed websites. By contrary, other new websites will be suspected as phishing attacks [15].

Possible phishing attacks on mobiles which can launch during the control transfers are discussed in [19]. An indicator for the applications identity upon the navigation bar of the system to show the currently running application or

the current web page was implemented in [20]. The personalized security indicators to mobile apps are proposed in [21]. However, the user-driven decision-making process is still needed.

The unified and trusted login user interface is used in another group of antiphishing techniques. A software keyboard which can be used safely for login input is provided in [22]. For the purpose of handling the credential, the hardware and software certificates that are used to confirm the login is proposed in [23]. However, these approaches require some modifications to the client application and the user effort. An antiphishing system for mobile platforms was presented in [24]. The work was continued in [4] to detect the persistent account registry phishing attacks. They used OCR technique, and their database needs to save every snapshot of the protected applications and webpages. Using the QR code in phishing attacks was demonstrated and analyzed in [25]. They combined the client-server architecture with a digital signature to perform an integrity checking and authentication. However, the work only focused on the QR code phishing attacks while the phishing malware was not considered. Phishing Detective [26] was created to identify whether or not a link in the user e-mail might send the user to a phishing page. However, the work was totally relied on the blacklist URL of Phish Tank database; it might not be able to satisfy other types of phishing attacks such as activity hijacking and repackaging attacks.

MP-Shield [27] is an Android application that aims to inspect the flow of IP packets between the origin and the destination of mobile user applications. Their work mainly emphasized on the monitoring URL for detection purposes. The types of phishing attack that can be mounted on mobile devices were identified in [19]. The authors conducted an analysis of ways in which the mobile applications and the web sites link to each other. The common control transfer on mobile and how phishing attacks can be mounted against the control transfer scenarios were discussed. The authors presented possible types of phishing attacks along with their legitimate behaviors as summarized in Table 1.

According to Table 1, the mobile sender means a mobile application that sends the user to a website or another mobile application, while the web sender means a website that sends the user to a mobile application or other web sites.

Our work will cover these attack models with ten groups of selected feature categories. Each phishing detection approach showed an acceptable detection accuracy while using different features. Unfortunately, majority of phishing detections may suffer the lack of features for efficient detection of phishing malwares. An optimized solution which used different kinds of features of Android applications to prevent the phishing and malware on Android smartphone is still needed. Our work will contribute to the finding of an optimal solution for mobile phishing detection in the sense of using the feature independently with various classifiers.

2.3. Case-Based Reasoning. Case-based reasoning (CBR) is a problem-solving approach that solves new problems by adapting or reusing old solutions that were used to solve

similar problems [28]. 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 [29]. 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 the 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 systems that are 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 [30]. The learning process of a case-based reasoning approach is shown in Figure 1.

Case-based reasoning system performs the learning process as follows:

- (1) Retrieving the most similar case or cases from the case base to the new problem
- (2) Reusing the previous solutions of the similar cases to solve the new problem
- (3) Revising the proposed solution (if necessary)
- (4) Retaining the solution of the new case for future problem solving

A new problem to the system is represented as a case and is compared with existing cases in the case base. The most similar case or cases are retrieved based on the similarity comparison of case representations. These retrieved cases are adapted (i.e., combined and reused) to propose a solution for the new problem. The suggested solution may need to be evaluated and corrected (i.e., revised) in some cases if it is not the best solution. This verified solution can be added back as a new case to the case base (i.e., retained) or as amendments to existing cases in the case base to be used in future problem solving [28].

3. Architecture Overview

A case-based reasoning model is proposed as an automatic adaptation of classifiers for mobile phishing detection. The information on how to design the case-based adaptive classification system is presented in this section. The proposed system consists of two main parts, including the application on Android smartphones and the detection system on the cloud environment. Figure 2 shows the overall system design.

As shown in Figure 2, the feature will be extracted from the Android application for the phishing detection process. The detailed information of features will be discussed in Section 3.1. Then, the extracted features will be sent to the cloud environment for phishing detection processes. As the

TABLE 1: Legitimate behaviors and their respective phishing attack techniques.

	Legitimate behavior	Respective attack techniques
Mobile sender	Social sharing, upgrades, game credits, opening a target in the browser, send user to embedded http page in browser that links to https login	Fake mobile login screen, task interception, scheme squatting, keylogging, URL bar hiding/spoofing, fake browser, using active network attack plus URL bar spoofing.
Web sender	Link to mobile e-mail or Twitter, payment via PayPal or Google checkout and user follows link from http to https	Website spoofs mobile app, task interception, scheme squatting, URL bar hiding/spoofing, active network attack plus URL bar spoofing.

FIGURE 1: Case-based reasoning approach.

FIGURE 2: Overall system design.

main objective of this work is to enhance the phishing detection processes, the detection will be performed on the static and dynamic feature from Android malware dataset (described in Section 4.1). The detailed process of feature extraction is out of this paper scope.

The contribution of our work starts on the receiving of the extracted features by the detection system. The first process is to retrieve the most similar case from the case base (which stored previous Android phishing detection approach along with the corresponding features). The case-retrieving process will be described in Section 3.3. The case base must be set up before the case-retrieving process. The case base setting up process is shown in Figure 3. The details of the case base setting up process are presented in the following section.

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.

3.1. Feature for Mobile Phishing. Existing antiphishing solutions on mobile environments were collected, and their features were extracted to identify a phishing attack. Under an Android environment, the features can be extracted from miscellaneous sources such as program entities and program outputs of the runtime monitoring. The list of frequently used features by existing antiphishing solutions can be classified into ten classes including Android components, Android API counts, API usage action, security-sensitive

FIGURE 3: Setting up case base.

data flow, hardware components, intent actions, permissions, shell command and strings, contents, and visual, and URLs. The details of each feature are given below.

- (1) Android components: a variety of component types with specific functionalities (e.g., components for providing GUIs, and others for running background services) are declared within an Android app's manifest, and these features are collected in [31–33].
- (2) API count: the number of invocations of a specific Android API method (e.g., the malicious apps access the location APIs twice and the telephony package 8 times) are collected in [4, 24, 27, 32].
- (3) API usage actions: APIs can be used to develop applications in Android platform and also misused by malicious purposes. There are many approaches to submit the web requests and to ex-filtrate the captured data via the API without the Internet permission. Some existing phishing detection works [27, 31, 32, 34] collect the API calls (e.g., API calls to access the sensitive data; API calls to access the network communications; API calls to send and receive the SMS messages; API calls to execute the external commands, and API calls frequently used for obfuscation).
- (4) Security-sensitive data flows: a few approaches for Android malware detection [31, 34, 35] use data flows between security-sensitive Android interfaces to determine if an app is malicious. Tracking this form of information is particularly useful for identifying privacy leaks.
- (5) Hardware components: the hardware components are listed in AndroidManifest.xml that is used in the app (e.g., to access the camera, an app needs to include android.hardware.camera feature) and these features are collected in [4, 36].
- (6) Intent actions: Android malwares are known to rely upon tracking of an Intent (e.g., whether a package is installed, or if a device has recently completed booting) to determine when to perform a malicious behavior. These features are used in [32, 36].
- (7) Permission: specific permissions, provided by Android to execute some risky operations, are acquired by Android malwares. These features are collected in [34, 37, 38].

- (8) Shell command and strings: the features of interested strings associated with malicious behaviors and potential risky shell commands are collected in [36, 39]. Some of the structural attributes of APK file, such as size of code, presence of zip file, binary file, and related information, are also included in this feature group.
- (9) Contents and visual: the main display channel for the deception of phishing is the web content which expresses the intention of the website. These features consist of the page elements, such as the page title, the submitted form, and the contained links. Some researchers also extract the logo icon and the contained pictures from the web page and use an image recognition algorithm to identify the phishing website [16–18].
- (10) URLs: web link features for phishing fraud is collected based on five criteria, including URL and Domain Identity, Security and Encryption, Source Code and Java script, Page Style and Contents, and Web Address Bar. These features are collected in [4, 13, 40].

3.2. 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 phishing 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 phishing detection system is shown in Table 2.

TABLE 2: Case description for mobile phishing 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 correct 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 4.

3.3. 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 phishing 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 neighbor method is applied to calculate the similarity score and the total similarity score of a potentially useful case.

3.4. 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 are stored in the case base) that is similar. The proposed case-based adaptive classification is shown in Figure 5. 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.

4. 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 phishing detection, an experiment is conducted using the feature sets (which has been described in Section 3.1). The experiment was conducted by running Weka 3.8 on a Laptop computer with core i7 processor, 8GB RAM, and Windows8.1 64 bit operating system. The cross-validation method is used as an evaluation technique to estimate the error rate efficiently and in an unbiased way by running repeated percentage splits. Firstly, the dataset is divided into 10 pieces. Each piece is used as a testing dataset in turn while the remaining 9 pieces together are used as a training dataset. We performed 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.

4.1. Dataset. The features are extracted from more than 10,000 Android malware samples which are collected from Android malware repositories including VirusShare [41], AndroZoo [42], Droid screening [43] and Reveal droid [44]. There are 76 extracted features of Android components, including 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. The information of the feature sets used in this experiment is shown in Table 3.

4.2. Machine Learning Classifiers. To detect and classify the phishing 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 [31, 32, 34, 35]. 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 [45], naive Bayes (NB) classifier with supervised discretization function, the

FIGURE 4: Case defining process (define a new case and store in case base).

FIGURE 5: Adaptive classification.

TABLE 3: Feature sets.

No	Feature sets	Number of features	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	Content_visual	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	

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.

4.3. Experimental Results and Analysis. The accuracy comparison of six classifiers on the 10 feature sets is shown in Table 4. The italicized values shown in Table 4 represent the maximum detection accuracy among six classifiers for each feature set. It can be seen that the accuracy of each

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

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 phishing 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.

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 methods 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. The italicized values shown in Table 6 represent the maximum detection accuracy of 5 cases among nine classifiers.

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 phishing detection heavily relies on the input features.

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 phishing detection model is shown in Table 7.

Performing the classification process on these large numbers of features takes a long runtime. The comparison of runtime to build the detection model on 6 base classifiers and 3 ensemble approaches before selecting the feature is shown in Table 8.

To reduce the detection time, some features may be omitted because the features may not provide a high impact on the result. Therefore, some experiments are conducted to select a set of effective features in order to reduce the number of required features.

4.4. Selecting the Features. Feature selection is necessary to reduce the dimension of the feature space. With the aim of

getting the benefits of performing a feature selection technique on a large data set such as reducing an overfitting issue, improving accuracy, and reducing a processing time, two feature selection techniques are performed in this experiment by comparing their results to get the optimized results. The process of selecting the features can be described by the following steps.

Let U be the universe of feature sets $U = \{D_1, D_2, \dots, D_v\}$. And, the dataset $D_i \in U$, with number v of attributes A , be $D_i = \{A_1, A_2, \dots, A_v\}$. Then, the attributes can be grouped into feature group FG_i as $FG_i = \{A_a, A_b, \dots, A_m\}$. Some attribute evaluation is performed and selected on the worth of each attribute, which becomes a selected feature set $FS_i = \{A_a, A_b, \dots, A_m\}$, where $FS_i \in FG_i$.

Two methods of feature selection techniques are used in this experiment to confirm the advantages of selecting the features in phishing detections. The first method is a correlation-based feature selection with a ranker search method that evaluates each attribute and lists the results in a ranked order. The worth of each attribute is evaluated by measuring the correlation (Pearson's) between it and the class [46].

Pearson's correlation coefficient is described in equation (1), where all variables have been standardized. The correlation between a composite and a class label is a function of the number of component variables (attributes) in the composite and the magnitude of the intercorrelations among them, together with the magnitude of the correlations between the attributes and the class label.

If the correlation between each of the attributes in a test and the class label is known, and the intercorrelation between each pair of attributes is given, then the correlation between a composite test consisting of the summed attributes and the class label can be predicted from the following equation:

$$r_{zc} = \frac{k\overline{r_{zi}}}{\sqrt{k + k(k-1)\overline{r_{ii}}}}, \quad (1)$$

where r_{zc} is the correlation between the summed attributes and the class label; k is the number of attributes; $\overline{r_{zi}}$ is the average of the correlations between the attributes and the class label; and $\overline{r_{ii}}$ is the average intercorrelation between attributes.

We get the ranked attributes listed with their corresponding class correlation. Some attributes, which owned no

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

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

TABLE 7: Tentative cases for mobile phishing detection system.

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

or less values on the class correlation measures, are eliminated. The resulting reduced feature sets are shown in Table 9.

The second method is an information gain attribute evaluation-based feature selection with a ranker search method. Information gain ratio evaluation is calculated by using the following equations. In the attribute evaluation processes, I index measures the impurity of D ; a data partition or a set of training tuples is calculated using

$$I(D) = 1 - \sum_{i=1}^m p_i^2, \quad (2)$$

where p_i is the probability that a tuple in D belongs to class C_i and is estimated by $(|C_{i,D}|/|D|)$. The sum is computed over m classes when I index considers a binary split for each attribute. First, the case where A is a discrete-valued attribute having v distinct values, $\{A_1, A_2, \dots, A_v\}$ occurring in D is considered. The expected information provided by that split is calculated by

$$I_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times I(D_j). \quad (3)$$

In this equation, D_j represents the observations that contain the j th attribute. The information gain of a binary split on attribute A is calculated by

$$\text{Gain}(A) = I(D) - I_A(D). \quad (4)$$

Information gain ratio attempts to correct the information gain calculation by introducing a split information value. The mathematical formulation for split information is provided in

$$\text{SplitInfo}_A(D) = - \sum_{j=1}^v \frac{|D_j|}{|D|} \times \log_2 \left(\frac{|D_j|}{|D|} \right). \quad (5)$$

This value represents the potential information generated by splitting the training dataset, D , into v partitions, corresponding to the v outcomes of a test on attribute A . The gain ratio is defined in

$$\text{Gain ratio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}_A(D)}. \quad (6)$$

The attribute with the maximum gain ratio is selected as the highest ranked attribute. The low-ranked attributes that provide a gain ratio less than 0.0003 are eliminated. After performing the two feature selection techniques on the dataset, the reduced feature sets are generated as shown in Table 9.

The same detection experiments are conducted with 9 classifications on each selected feature set. The detection results of 5 cases on selected feature sets are described in Tables 10 and 11. In this experiment, 9 classification approaches with their related parameters are set up as the same as that of previous experiments (described in Section 4.2).

According to the results of the reduced datasets with a correlation attribute evaluation method shown in Table 10, the classification approaches with the best detection accuracy are slightly changed in 2 cases (feature patterns 3 and 4). Feature pattern 3 is a combination of API count, API usage, Intent, and Hardware. The italicized values shown in Table 10 represent the maximum detection accuracy of 5 cases among nine classifiers. The highest detection accuracy is now provided by ensembles with AVG and MAJ final answer methods, while the highest detection accuracy is provided by ensembles with the AVG final answer method when full feature set is used. The detection accuracy is slightly increased for most classifiers in feature pattern 4 which is a combination of flows and Intents features.

According to the results shown in Table 11 of the reduced datasets with an information gain attribute evaluation

TABLE 8: Runtime comparison of 5 scenarios on 9 classification approaches (in seconds).

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
01	4.43	18.63	0.01	1.93	0.64	3.01	29.87	29.52	26.16
02	4.54	31.80	0.0001	2.57	0.66	39.24	76.20	76.35	75.90
03	9.44	58.80	0.01	7.22	1.14	18.94	95.18	96.09	97.41
04	12.09	148.32	0.0001	5.28	1.39	6.25	174.4	174.6	174.61
05	17.09	167.14	0.01	7.86	1.93	3.61	203.62	205.50	203.51

TABLE 9: Information of selected feature sets for 5 cases.

Case ID	Feature combination pattern	Features before feature selection	Features selected by Pearson's correlation	Features selected by information gain
01	Pattern 1	112	96	100
02	Pattern 2	139	114	120
03	Pattern 3	220	180	185
04	Pattern 4	529	164	265
05	Pattern 5	610	227	250

TABLE 10: Detection accuracy of 5 cases after correlation attribute evaluation feature selection.

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
01	95.87	93.10	95.45	92.47	89.42	91.61	95.31	95.32	92.82
02	94.68	91.53	94.04	90.18	86.44	89.28	94.24	94.18	91.33
03	96.37	92.70	95.60	94.90	90.69	92.57	96.38	96.38	94.37
04	90.73	86.51	90.45	88.51	81.55	87.89	90.73	90.72	88.64
05	95.38	89.54	94.37	93.96	92.28	91.61	95.68	95.69	92.72

TABLE 11: Detection accuracy of 5 cases after information gain attribute evaluation feature selection.

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
01	95.96	93.10	95.62	92.54	89.42	91.66	95.37	95.37	92.83
02	94.66	91.53	94.16	90.15	86.44	89.19	94.19	94.12	91.30
03	96.38	92.59	95.55	95.02	90.69	92.61	96.45	96.45	94.34
04	90.52	86.36	90.24	88.70	81.55	87.86	90.77	90.76	88.69
05	95.46	89.44	94.50	93.98	92.28	91.56	95.80	95.79	92.82

method, the detection accuracy is increased in 4 cases (feature patterns 1, 3, 4, and 5). The italicized values shown in Table 11 represent the maximum detection accuracy of 5 cases among nine classifiers. Moreover, the classification approaches which produced the best detection accuracy are changed in 3 cases (feature patterns 3, 4, and 5). That is, an ensemble with AVG final answer finding method provides the best accuracy for feature patterns 3, 4, and 5.

The detection accuracy percentages of 5 cases by using different algorithms are comparatively described in Figure 6. This figure represented the detection results from Tables 6, 10, and 11. Each case is represented in 3 situations such as no features selection, after correlation attribute evaluation feature selection, and after information gain attributes evaluation feature selection. There are 15 points in the figure, representing the 5 cases with 3 conditions. The best classifier for case 01 and case 02 is J48 classifier while ensemble classifier AVG is the best one for case 03, case 04, and case 05. The cases with the best algorithm are used in the case-based reasoning detection method.

With the aim of highlighting the performance of feature selection techniques, the runtime results of reduced feature sets are collected as described in Tables 12 and 13. The

information gain attribute evaluation method results in a large number of features than the correlation attribute evaluation method. The runtime of the information gain attribute evolution method is also slightly larger than that of the correlation attribute evaluation method.

The runtime on 5 cases by selecting the features are showed in Figure 7. This figure compared the runtime from Tables 8, 12, and 13. There are 15 points in the figure representing the 5 cases with 3 conditions.

Selecting the features with the information gain attribute evaluation approach is applied on our feature sets to improve our model for better accuracy and efficiency. The percentages of detection accuracy on 4 feature patterns are improved as shown in Table 11 while the performances of the detection on all feature patterns are improved as shown in Table 13. Table 14 shows the comparison of accuracy and efficiency of full feature sets and reduced feature sets of our proposed adaptive model. The italicized values shown in Table 14 represent the accuracy values when a reduced feature set is used, and the accuracy values are improved over their counterpart when a full feature set is used.

The phishing malware detection task is an imbalanced classification problem. That is, there are two classes to be

FIGURE 6: Accuracy comparison of 9 classifiers on 5 cases before and after feature selection.

TABLE 12: Runtime comparison after correlation attribute evaluation feature selection (seconds).

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
01	3.94	18.36	0.01	1.97	0.64	3.38	27.45	26.92	27.18
02	3.84	25.35	0.0001	2.43	0.50	39.46	72.07	72.12	71.19
03	8.06	45.85	0.01	7.20	1.03	19.55	83.10	83.52	83.34
04	5.60	44.75	0.0001	5.15	0.56	6.27	61.95	61.99	62.00
05	8.84	69.88	0.0001	7.65	1.00	3.20	90.23	90.06	90.45

TABLE 13: Runtime comparison after information gain attribute evaluation feature selection (seconds).

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
01	4.05	20.25	0.01	1.63	0.54	3.02	29.02	27.49	27.27
02	3.86	29.43	0.001	2.35	0.56	31.96	67.63	66.52	66.40
03	9.77	55.46	0.01	6.31	0.95	17.06	87.31	90.63	90.69
04	6.83	87.36	0.01	2.25	0.93	6.76	102.86	93.21	93.15
05	8.42	104.52	0.001	5.37	1.21	3.95	111.80	107.53	108.07

identified, including phishing and benign, with one category representing the overwhelming majority of the data points. In these cases, the positive class “phishing” is greatly outnumbered by the negative class. These types of problems are examples of the fairly common case in the data science when the accuracy is not a good measure for assessing the model performance. Intuitively, proclaiming all data points as negative in the phishing detection problem is not helpful and, instead, we should focus on identifying the positive cases.

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:

$$\begin{aligned} \text{Accuracy} &= \frac{TP + TN}{TP + FP + TN + FN}, \\ \text{Precision} &= \frac{TP}{TP + FP}, \\ \text{Sensitivity} &= \frac{TP}{TP + FN}. \end{aligned} \quad (7)$$

FIGURE 7: Runtime comparison of 9 classifiers on 5 cases before and after feature selection.

TABLE 14: Accuracy and efficiency of proposed adaptive model.

Case ID	Adaptive (before)	Adaptive (after)	Accuracy (before)	Accuracy (after)	Runtime (before)	Runtime (after)
01	J48	J48	95.93	95.96	4.43	4.05
02	J48	J48	94.72	94.66	4.54	3.86
03	AVG	AVG, MAJ	96.43	96.45	95.18	87.31, & 90.63
04	AVG, MAJ	AVG	90.64	90.77	174.4, & 174.6	102.86
05	MAJ	AVG	95.69	95.80	205.50	111.80

True positive (TP) is the amount of correct positive prediction; false positive (FP) is the incorrect positive prediction; true negative (TN) is the amount of correct negative prediction; and false negative (FN) is the amount of incorrect negative prediction. These four outcomes form the confusion matrix as shown in Figure 8.

The evaluation of effectiveness on our proposed model by means of accuracy, precision, and sensitivity is described in Table 15. According to the results shown in Table 15, our adaptive model achieves a good detection accuracy for the phishing features. Meanwhile, the performance of all the classifiers gets an acceptable precision and sensitivity ratio. According to the previous experiments, our adaptive phishing detection model using case-based reasoning can perform well on the diversely distributed features.

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

FIGURE 8: Confusion matrix.

TABLE 15: 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

5. Conclusions

An adaptive mobile phishing detection model based on a variation of input feature patterns using a case-based reasoning (CBR) technique is proposed in this work. 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 phishing 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 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. In total, there are 9 classifiers. Due to the involvement of efficient features in the dataset and the uses of multiple classifiers, the efficiency degradation happened. To overcome this hurdle, 2 feature selection techniques are applied on the dataset in order to reduce the size of the features which is the size of the input to the classifiers. The two feature selection techniques used are information gain attribute evaluation method and Pearson's correlation coefficient attribute evaluation method. By addressing the optimal selection of the suitable classifier to the incoming features using a case-based reasoning approach, the proposed mobile phishing detection model could provide an accuracy improvement with an acceptable runtime increment.

Data Availability

The dataset of the features used in this research is available from the authors upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the Higher Education Research Promotion and the Thailand's Education Hub for Southern Region of ASEAN Countries Project Office of the Higher Education Commission.

References

- [1] W. Paul, H. A. Manolian, and S. Lapper, "Thinking digital in industry 4.0," Deloitte Insights, September 2018, <https://www2.deloitte.com/insights/us/en/focus/industry-4-0/digital-leaders-in-manufacturing-fourth-industrial-revolution.html>.
- [2] "Spam and phishing in Q2 2018," Securelist-Kaspersky Lab's Cyberthreat Research and Reports, 2018.
- [3] Proofpoint Security Awareness, "2019 state of the phish report," March 2019, <https://www.wombatsecurity.com/state-of-the-phish>.
- [4] L. Wu, X. Du, and J. Wu, "Effective defense schemes for phishing attacks on mobile computing platforms," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 8, pp. 6678–6691, 2016.
- [5] M. Moghimi and A. Y. Varjani, "New rule-based phishing detection method," *Expert Systems with Applications*, vol. 53, pp. 231–242, Jul. 2016.
- [6] Baunfire.com and SparkCMS, "APWG phishing attack trends report-4Q 2018," Anti-Phishing Working Group, March 2019, <https://www.antiphishing.org/resources/apwg-reports/>.
- [7] R. Basnet, S. Mukkamala, and A. H. Sung, "Detection of phishing attacks: a machine learning approach," in *Soft Computing Applications in Industry*, B. Prasad, Ed., pp. 373–383, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
- [8] A. K. Jain and B. B. Gupta, "Comparative analysis of features based machine learning approaches for phishing detection," in *Proceedings of the 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 2125–2130, New Delhi, India, March 2016.
- [9] F. Toolan and J. Carthy, "Phishing detection using classifier ensembles," in *Proceedings of the 2009 eCrime Researchers Summit*, pp. 1–9, Tacoma, WA, USA, October 2009.
- [10] H. S. Hota, A. K. Shrivastava, and R. Hota, "An ensemble model for detecting phishing attack with proposed remove-replace feature selection technique," *Procedia Computer Science*, vol. 132, pp. 900–907, 2018.
- [11] *A Comparative Study of Phishing Websites Classification Based on Classifier Ensembles*, ResearchGate, Berlin, Germany, 2019, https://www.researchgate.net/publication/325483941_A_Comparative_Study_of_Phishing_Websites_Classification_Based_on_Classifier_Ensembles.
- [12] W. Wang, Y. Li, X. Wang, J. Liu, and X. Zhang, "Detecting Android malicious apps and categorizing benign apps with ensemble of classifiers," *Future Generation Computer Systems*, vol. 78, pp. 987–994, 2018.
- [13] A. Aleroud and L. Zhou, "Phishing environments, techniques, and countermeasures: a survey," *Computers and Security*, vol. 68, pp. 160–196, 2017.
- [14] H. Shahriar, T. Klantic, and V. Clincy, "Mobile phishing attacks and mitigation techniques," *Journal of Information Security*, vol. 6, no. 3, pp. 206–212, 2015.
- [15] T. M. Mahmoud and A. M. Mahfouz, "SMS spam filtering technique based on artificial immune system," *International Journal of Computer Science Issues*, vol. 9, no. 1, pp. 589–597, 2012.
- [16] J. W. Yoon, H. Kim, and J. H. Huh, "Hybrid spam filtering for mobile communication," *Computers and Security*, vol. 29, no. 4, pp. 446–459, 2010.
- [17] C. H. Hsu, P. Wang, and S. Pu, "Identify fixed-path phishing attack by STC," in *Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference*, pp. 172–175, Perth, Australia, September 2011.
- [18] E. Medvet, E. Kirda, and C. Kruegel, "Visual-similarity-based phishing detection," in *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks*, Istanbul, Turkey, September 2008.
- [19] A. P. Felt and D. Wagner, *Phishing on Mobile Devices*, University of California, Berkeley, CA, USA, 2011.
- [20] A. Bianchi, J. Corbetta, L. Invernizzi, Y. Fratantonio, C. Kruegel, and G. Vigna, "What the app is that? Deception and countermeasures in the android user interface," in *Proceeding of the 2015 IEEE Symposium on Security and Privacy*, pp. 931–948, San Jose, CA, USA, May 2015.
- [21] C. Marforio, R. J. Masti, C. Soriente, K. Kostinen, and S. Capkun, "Personalized security indicators to detect application phishing attacks in mobile platforms," February 2015, <http://arxiv.org/abs/150206824>.
- [22] D. Liu, E. Cuervo, V. Pistol, R. Scudellari, and L. P. Cox, "ScreenPass: secure password entry on touchscreen devices," in *Proceeding of the 11th Annual International Conference on*

- Mobile Systems, Applications, and Services*, pp. 291–304, Taipei, Taiwan, June 2013.
- [23] D. Liu and L. P. Cox, “VeriUI: Attested Login for Mobile Devices,” in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, Santa Barbara, CA, USA, February 2014.
 - [24] L. Wu, X. Du, and J. Wu, “MobiFish: A lightweight anti-phishing scheme for mobile phones,” in *Proceedings of the 2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–8, Shanghai, China, August 2014.
 - [25] V. Mavroeidis and M. Nicho, “Quick response code secure: a cryptographically secure anti-phishing tool for QR code attacks,” in *Computer Network Security*, pp. 313–324, 2017.
 - [26] “Phishing detective-apps on Google Play,” March 2018, <https://play.google.com/store/apps/details?id=com.rsofr.android.phishingdetectiveads>.
 - [27] G. Bottazzi, E. Casalicchio, D. Cingolani, F. Marturana, and M. Piu, “MP-Shield: A framework for phishing detection in mobile devices,” in *Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pp. 1977–1983, Liverpool, UK, October 2015.
 - [28] M. M. Richter and R. O. Weber, *Case-Based Reasoning*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
 - [29] S. Craw, N. Wiratunga, and R. C. Rowe, “Learning adaptation knowledge to improve case-based reasoning,” *Artificial Intelligence*, vol. 170, no. 16–17, pp. 1175–1192, Nov. 2006.
 - [30] 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 Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 4, pp. 421–434, Jul. 2011.
 - [31] S. Arzt, “FlowDroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps,” in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 259–269, New York, NY, USA, June 2014.
 - [32] L. Li, A. Bartel, T. F. Bissyande et al., “IccTA: detecting inter-component privacy leaks in android apps,” in *Proceedings of the 37th International Conference on Software Engineering*, vol. 1, pp. 280–291, Piscataway, NJ, USA, May 2015.
 - [33] Obfuscation-resilient, efficient, and accurate detection and family identification of android malware—semantic scholar,” March 2018, <https://paper/Obfuscation-Resilient%2C-Efficient%2C-and-Accurate-and-Garcia-Hammad/959093db69abc3b0fb4f7acc696a7f6ef39d0e23>.
 - [34] W. Enck, “TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones,” *Transactions on Computer Systems*, vol. 32, no. 2, 2014.
 - [35] M. I. Gordon, D. Kim, J. Perkins, L. Gilham, N. Nguyen, and M. Rinard, “Information-flow analysis of android applications in DroidSafe,” in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2015.
 - [36] D. Arp, M. Spreitzenbarth, H. Gascon, and K. Rieck, “Drebin: effective and explainable detection of android malware in your pocket,” in *Proceedings of the 2014 Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2014.
 - [37] N. Peiravian and X. Zhu, “Machine learning for android malware detection using permission and API calls,” in *Proceedings of the 2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, pp. 300–305, Herndon, VA, USA, November 2013.
 - [38] V. Avdiienko, K. Kuznetsov, A. Gorla et al., “Mining apps for abnormal usage of sensitive data,” in *Proceedings of the 37th International Conference on Software Engineering*, vol. 1, pp. 426–436, Florence, Italy, May 2015.
 - [39] H. V. Nath and B. M. Mehtre, “Static malware analysis using machine learning methods,” in *Recent Trends in Computer Networks and Distributed Systems Security*, pp. 440–450, 2014.
 - [40] N. Abura’ed, H. Otrouk, R. Mizouni, and J. Bentahar, “Mobile phishing attack for Android platform,” in *Proceedings of the 2014 10th International Conference on Innovations in Information Technology (IIT)*, pp. 18–23, Abu Dhabi, UAE, November 2014.
 - [41] VirusShare.com, <https://virusshare.com>.
 - [42] 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*, pp. 468–471, Austin, TX, USA, May 2016.
 - [43] J. Yu, Q. Huang, and C. Yian, “DroidScreening: a practical framework for real-world Android malware analysis,” *Security and Communication Networks*, vol. 9, no. 11, pp. 1435–1449.
 - [44] Joshua/Revealdroid—Bitbucket, <https://bitbucket.org/joshuaga/revealdroid/src/master/>.
 - [45] S. Kyaw Zaw and S. Vasupongayya, “Revealing the important features of mobile phishing,” in *Proceedings of the 13th International Conference on Knowledge, Information and Creativity Support Systems (KICSS 2018)*, pp. 222–226, Pattaya, Thailand, November 2018.
 - [46] M. A. Hall and L. A. Smith, “Feature subset selection: a correlation based filter approach,” *Progress in Connectionist-based Information Systems*, vol. 2, pp. 855–858, 1997.