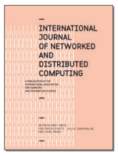
International Journal of Networked and Distributed Computing
Vol. **8**(**3**); June (2020), *pp.* 152–161
DOI: https://doi.org/10.2991/ijndc.k.200515.001; ISSN 2211-7938; eISSN 2211-7946
https://www.atlantis-press.com/journals/ijndc



Research Article

Enhancing Case-based Reasoning Approach using Incremental Learning Model for Automatic Adaptation of Classifiers in Mobile Phishing Detection

San Kyaw Zaw, Sangsuree Vasupongayya*

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

ARTICLE INFO

Article History

Received 09 December 2019
Accepted 02 February 2020

Keywords

Incremental learning model
adaptive phishing detection
case-based reasoning
concept drift
mobile phishing

ABSTRACT

This article presents the threshold-based incremental learning model for a case-base updating approach that can support adaptive detection and incremental learning of Case-based Reasoning (CBR)-based automatic adaptable phishing detection. The CBR-based adaptive phishing detection model detects the phishing with the most suitable machine learning technique and this appropriate detection approach is endorsed by CBR technique. In such a way, the adaptive phishing detection model can address the concept drift. The threshold-based incremental learning model for a case-base updating approach will address the comprehensiveness of the knowledge in the case-base to support an incremental learning. The prototype system of our model is evaluated using nine testing feature groups of more than 20,000 phishing instances. The result shows that our adaptive phishing detection system maintains the detection accuracy while learning the new cases incrementally. The evaluation results indicate that our approach is more flexible to address the concept drift with a stable accuracy and a better performance.

1. INTRODUCTION

Nowadays, millions of mobile phone users over the world are put at risk by phishing while more than 3.8 billion smartphones are estimated to be used in 2020 [1]. As a consequence, the security of these devices becomes a top priority. Moreover, mobile devices become the primary means of communication and information access [2]. Thus, in our prior work [3], some analyses on the literatures of phishing detection are performed and identified the important features for the mobile phishing detection. Then, the adaptive mobile phishing detection model is proposed in another prior work [4] by using a Case-based Reasoning (CBR) approach. In our previous work [4], the experiments were conducted to demonstrate the design decision of our proposed model and to verify the performance in handling the concept drift. However, the main challenge faced by the CBR approach is learning a new case in order to adapt the system to a new phishing pattern. The mismatching input features with the existing cases in the case-base was lacking in our prior work [4]. In this work, the incremental learning model for the adaptation to the new examples to the case-base is proposed.

The majority of researches have addressed the mobile phishing attacks by using machine learning approaches. The machine learning approaches construct the mobile phishing detection solution with the example data or the past experience to improve the performance and the accuracy [5] with some additional challenges. In other word, the accuracy of most machine learning algorithms depend on their corresponding features.

*Corresponding author. Email: vsangsur@coe.psu.ac.th

In our prior work [4], an adaptive phishing detection model is proposed to perform a detection with a stable accuracy on various mobile phishing features. The effectiveness of that work was showed with a small prototype system containing five sample cases which are the randomly combined features patterns. The feature patterns of that previous work included ten features groups which are commonly extracted and used in some existing phishing detection works. Some experiments have also been conducted to analysis the possible matching of feature to classifiers on detection in order to handle the concept drift. The almost stable detection accuracy on variety of feature patterns confirmed the effectiveness of the proposed solution in our prior work [4].

Since the adaptation of classifiers according to the incoming feature patterns is a key manner for improving the detection, an incremental learning approach becomes an essential ingredient for any case-based reasoning system. Thus, the detail information of the case-base updating process which includes retraining the solution of the new case, handling the mismatched features and learning a new case from the stored features, will be analyzed in this work. Phishing in this work is not focusing only the information stealing but also the mobile device hijacking as well. Therefore, many malicious features on the mobile devices will also include in this work.

A comparative analysis on the detection performance of this work and that of our prior work showed that the proposed incremental learning approach maintained stably the detection accuracy and the performance while learning totally new features and less matching features to the case-base. The good detection performances can be observed include the detection accuracy and the acceptable precision and sensitivity scores.

This paper is structured as follows. In Section 2, some related works to the phishing detection, the theoretical background of a CBR approach, and the details of our previous work are discussed. In Section 3, the proposed incremental learning model for enhancing the case-based reasoning is explained. The design decision of the proposed enhancing system is explained. In Section 4, the proposed incremental learning model is evaluated with a number of Android apps dataset to show how it can support the model learning ability for handling the concept drift. The conclusion is given in Section 5.

2. RELATED WORKS AND BACKGROUND THEORY

In this section, the use of machine learning algorithms and their related features that used in existing literatures on phishing detections are reviewed. The case-based reasoning approach that used in our previous phishing detection system is also presented. Then, the information on our previous adaptive phishing detection model is outlined. Moreover, the learning process to support the up-to-date case-base that is lacking in our prior work is also discussed.

2.1. Related Works

Various applications of machine learning techniques have been applied to the phishing detection problem. Several machine learning approaches are used for combating phishing attacks. These approaches include Support Vector Machine (SVM), Bayesian additive regression trees, Random Forests (RF), classification and regression trees, Logistic Regression (LR), and Artificial Neural Networks (ANN) [6]. A nonlinear regression strategy was used in Rao and Pais [7] for detecting a phishing web site. They preferred the use of a harmony search and support vector machine meta-heuristic algorithms for training the system. The authors Basnet et al. [8] and Jain and Gupta [9] used specific detection algorithms with selected feature patterns in their specific domain.

Two machine learning aided approaches for static analysis of Android malware were proposed in Chin et al. [10]. The first approach was based on the permissions and the other was based on the source code analysis utilizing a bag-of-words representation model. A phishing detection and mitigation approach with software-defined networking was proposed to identify the phishing activities through an e-mail and a web-based communication [11]. A phishing detection model based on deep belief networks was presented in Yi et al. [12] that works on real IP flows from Internet Service Provider (ISP).

Each of these phishing detection approaches showed the acceptable detection accuracy, the detection accuracy and performance are limited to the use of features and machine learning techniques [13]. Again, some ensemble method in the form of a combination of multiple classifiers was used in Toolan and Carthy [14], Hota et al. [15] and "A Comparative Study of Phishing Websites Classification Based on Classifier Ensembles" [16]. Although, these ensemble method-based detection system could improve the accuracy as well as the efficiency, they suffered from concept drift issues because they could not support the automatic adaptability to the various input features.

A phishing detection system using a case-based reasoning was presented in Abutair et al. [17], by using the features of phishing Uniform Resource Locator (URLs) as the case and classifying the phishing URL. Their work concentrated on the URL feature extraction and the keyword similarities in the main domain name and the remaining keywords in the URL such as the path, subdomains, and query part. CBR was used in Abutair et al. [17] as a classifier for the new incoming web link URL to detect the phishing website. In our previous work [4], CBR was used to adaptively select the most suitable classifier among existing classifiers to solve the current case.

2.2. Case-based Reasoning

In the CBR technique [18], the new problems were solved by re-usable or adapted solutions which were previously used in the past. Under the situation of solving the unfamiliar problem in the domains that are not well understood, CBR can perform well, by using the knowledge on what worked in the past. The experiences from the previously solved problems are encoded as cases and each case contains the feature characteristic of the problem and the corresponding solution. The case-base maintains the collection of cases and it can be used as the knowledge base to handle the new problems. Thus, CBR is used to handle the concept drift, for endorsing the optimized machine learning in our previous work [4].

In our CBR-based automatic adaptable mobile phishing detecting model, the feature pattern of the mobile app and the corresponding machine learning algorithm which can support the optimized accuracy and efficiency will be encoded into cases and stored in our case-base. For a new feature input, the case-base will endorse the best machine learning scheme for making a detection. The general structure of the CBR as a cyclical process [19] which can be divided into the sub processes is represented in Figure 1.

The four main sub-processes are existed in a case-based reasoning process:

(i) Searching the most similar case to the current problem.

(ii) Re-applying the similar case solution to solve the current problem.

(iii) Feeding back the proposed solution to the system, if requires.

(iv) Maintaining the solution for future uses.

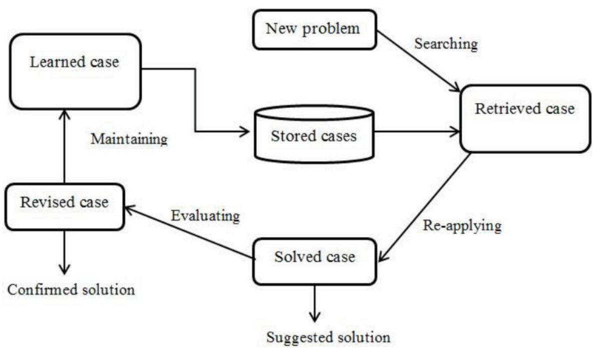


Figure 1 | Case-based reasoning process.

Basically, a new problem is symbolized as a case, and it is compared with the stored cases in the case-base. Then, the most similar case or cases are selected according to the comparative results of the case representation features. These cases will be combined or reused to suggest a possible solution for the new problem. The suggested solution is evaluated and corrected if a revision is needed. The confirmed solution will be maintained by adding it back as a new case to the case-base or as an amendment to existing cases in the case-base for a future problem.

In our previous work [4], various machine learning algorithms are applied to the problem of mobile phishing detections and the CBR technique is applied. Thus, an approach for retaining the solution of the new case is proposed in this work to enhance the previous work in adapting to the new case by learning it and combine it into our system in a systematical way. Incremental learning can be defined as learning a concept incrementally by processing predefined training examples in a one-by-one manner [20].

2.3. Proposed Adaptive Phishing Detection Model

An adaptive mobile phishing detection model that will work on a variation of input feature patterns using a CBR technique is proposed in our prior work [4]. The overall system design of the proposed adaptive phishing detection system is shown in Figure 2.

The extracted features from the active Android apps are fed to the detection system to find the most similar case in the case-base. When the extracted features from the active Android apps are received by the detection system, the first process is selecting the most similar case from the case-base and retrieving the similar case. The details of the case-base setting up process were presented in our previous work [4]. According to the information of the retrieved case, the corresponding classification techniques will be performed on the extracted features by the adaptive classification.

When the extracted features of the active Android apps do not match with the features of the stored cases in the case-base, the incremental learning process will be performed on the extracted features using multiple classifiers. Then, the optimal classification technique with the maximum detection accuracy is selected and the corresponding information including the features with the corresponding classifiers is stored in the case-base. Then, the result of the active Android application classification will be displayed to the user.

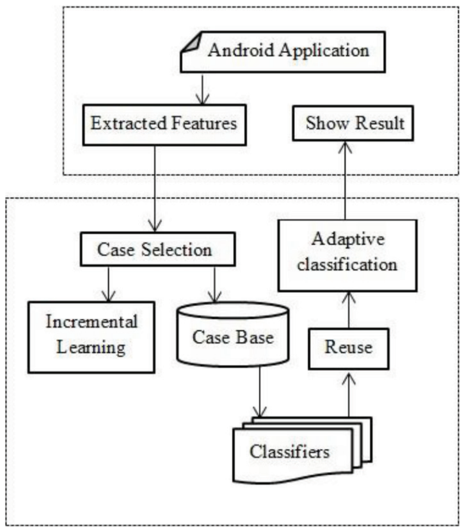


Figure 2 | System design of adaptive phishing detection.

2.4. Incremental Learning to Update the Case-base

Our prior work proposed to create a phishing detection system based on the CBR approach to automatically adapt the classifiers depending on the features input patterns. Combining the good performance of all classification methods appropriately, the proposed phishing detection system could provide the best performance by addressing the optimal selection of the appropriate classifier to the input features using a case-based reasoning approach. However, the details of the learning process to update the case-base for handling the concept drift were lacking in our prior work [4]. In this work, the threshold-based incremental learning model for the CBR-based adaptive phishing detection is proposed and evaluated.

The incremental learning model for the case-based updating process constitutes the learning process for the new cases. The learning process will be launched in two possible cases; first, the learning process will be launched when the output from the reused process meets the threshold acceptable point; second, the learning process will be launched when an uncertain output is observed [21]. The solution have to be revised and modified to become a confirmed solution and the appropriate retaining or adding to the case-base are included for further problems [22]. The case-base updating process involves two steps; the proposed solution is evaluated and repaired, if needed. The goodness and competence of the proposed solution to the case-base is judged in the evaluation step.

Generally, there are two categories of learning techniques in the case-base updating policy. The first one is a continuous case-base updating with training examples that have been misclassified by the system. The second one is a periodic retraining of the classifier to reselect the features. Performing these two approaches of learning, the system will be more predictive in finding the most appropriate detection algorithm.

The evaluation process can be performed in a number of different methods such as based on the results of the simulations when applying the solution. The evaluation outcomes could suggest the further adaptation to the case-base or the repairing of the proposed solution. When the further adaptation is performed on the solution, the revised case will be added to the case base. Thus, the case-base updating process is the ongoing argument of the case-base with the revised case thereby allowing the system to learn. Successful new solutions are added to the case-base memory to facilitate any similar problem in the future and failures can be added to avoid repeating the same mistakes. Increasing the number of example situations in the case-base means that the system covers more domains and will work better. In this work, updating the solution of the new case is addressed for our case-based reasoning approach to enhance the automatic adaptation of classifiers in a mobile phishing detection domain.

3. INCREMENTAL LEARNING FOR ADAPTIVE PHISHING DETECTION MODEL

The main objective of the case-based adaptive classification in our prior work [4] is to assign a suitable classification technique on the new case with the new extracted feature set, by calculating the similarity score of the sets of features of the stored case in the case-base. The main objective of the incremental learning model proposed in this work is to support the adaptability of the CBR-based adaptive phishing detection system with an incremental learning. The overall system design of the incremental learning-based adaptive phishing detection model is shown in Figure 3.

The incremental learning model for the adaptive phishing detection system using the CBR consists of two main components containing retrieving and reusing the suitable solution of the previously stored cases (Adaptive Classification), and finding the optimize solution on the new case for future learning process (Incremental Learning). According to our system design, when the extracted features of the new Android apps arrive at the system, the suitable classification technique selection is performed based on the similarity threshold of the extracted features. If the similarity score of the extracted feature set from the active Android application does not reach the acceptable threshold to proceed with the case retrieving process, the incremental learning process will be performed to find an optimal solution on the extracted feature set. Thus, multiple classifiers are processed with 12 classification techniques on the new extracted features set. The 12 classification techniques include six individual classifiers, a stacking ensemble with a logistic regression as a meta-classifier and three voting ensembles on multiple classifiers, and a multilayer perceptron Neural Network. These stacking and voting ensembles will work on the multiple answers that resulted from six individual classifiers. The final process is taking the maximum detection accuracy from all 12 classification results. Then, the new features with the corresponding solution will be stored in the case-base for the incremental learning process.

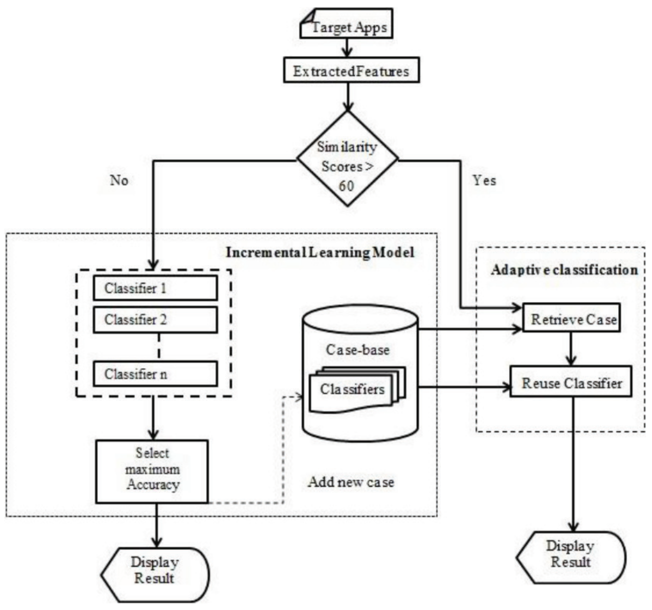


Figure 3 | Incremental learning based adaptive phishing detection model.

3.1. Feature Extraction

The feature extraction process will extract the features of the active Android application for the detection process. Features can be extracted and collected from various sources in the Android environment by using static analysis techniques (e.g., AndroidManifest.xml and class files) and dynamic analysis techniques. For example, the required permissions and the sensitive APIs will be extracted through the static analysis, whereas the dynamic behaviors will be extracted through the dynamic analysis.

Under the static analysis technique, the installation file (i.e., *.apk file) of each Android app are decompressed and the resulted AndroidManifest.xml and Classes.dex files are parsed. By parsing the Android Manifest.xml file, the permissions required by the apps are obtained. By parsing the classes.dex file with the disassembler, the usages of the API functions are extracted.

Under the dynamic analysis technique, the sandboxing software (i.e., CuckooDroid Sandbox) needs to be installed and processed on each apps for automating analysis of the suspicious files. The monitoring process will take place by hooking the chosen Dalvik API calls for extracting the behavioral information such as file input/output, short message services, mobile phone calls, and information leaks. Feature extraction process performed by the system is discussed in previous work [4]. When our phishing detection system received the extracted features of android apps, the similarity scores of these features are calculated by our system.

3.2. Feature Selection

About 1065 frequently used mobile phishing features were used in our prior work [4]. These features were extracted from more than 10,000 Android samples, collected from public Android malware repositories. The mobile phishing features are classified into 10 classes including Android components, Android API counts, API usage action, security sensitive data flow, hardware components, intent actions, permissions, shell command and strings, contents and visual, and URLs. Feature selection techniques on these ten feature sets are performed in this work to reduce the feature space dimension, to reduce the over fitting issue as well as to improve the accuracy and efficiency of the detection. The comparative analysis of two feature selection methods including correlation-based feature selection and information gain based feature selection, are conducted in our prior work [4].

In this work, the information gain attribute evaluation-based feature selection with a ranker search method is performed on the ten feature sets. Then, the highest ranked attributes with the maximum gain ratio are selected and the lower ranked attributes are excluded for conducting the experiments in this work. After performing the information gain attribute evaluation based feature selection technique on the dataset, the total number of features in the reduced features sets are 552 features.

3.3. Case Retrieval with Cosine Similarity Calculation

Finding the stored cases closet to the current problem in the case-base is the case retrieving process. The closet case gets the opportunity to be a suitable solution for the new problem. The finding process of the most suitable case was believed to be the core of CBR [22]. One challenge related to the case retrieving process is the computation of the similarity score during the case retrieving process. The usefulness of a retrieved case in a new problem solving determines the effectiveness of the similarity measurement. In this work, the total similarity scores of the potential cases are calculated with the cosine similarity measure. The cosine similarity calculates the similarity metric between two vectors of an inner product space by comparing the input target case to the cases in the case-base. It is measured by the cosine of the angle between two vectors and determined whether two vectors are roughly pointing in the same direction. This technique is often used to measure the document similarity in text analysis problem [23].

The similarity between a stored case vector C and the test set as a query vector T is calculated using the cosine similarity function where the cases deal with the feature attributes name (textual information). This ratio is defined as the cosine of the angle between the vectors, within the values between 0 and 1 and can be calculated by Equations (1) and (2).

$$\mathrm{Sim}(C,T) = \frac{C.N}{\|C\|\|N\|} \tag{1}$$

$$\frac{C.T}{\|C\|\|T\|} = \frac{\sum_{i=1}^{n} C_i \times T_i}{\sqrt{\sum_{i=1}^{n} C_i^2} \times \sqrt{\sum_{i=1}^{n} T_i^2}} \tag{2}$$

The closet measurements of each feature of the stored cases to the new case are formed the similarity metric. These closing measurements are presented as the similarity scores and the score is assigned depending on the difference between the two features. That is, the closer the features the higher the similarity scores. The features according to their similarity score are ranked for the most relevance to the current problem to the least relevance to the current problem. The cases with the highest similarity values will be retrieved. The prediction result of the new problem can then be derived from the solution of these most similar cases.

4. EVALUATION OF INCREMENTAL LEARNING-BASED ADAPTIVE PHISHING DETECTION MODEL

The threshold-based incremental learning model for the case-base updating approach that can support the adaptive detection and the incremental learning in the CBR-based automatic adaptable phishing detection model is presented in this work. The threshold-based incremental learning model for the case-base updating approach will address the comprehensiveness of the knowledge in the case-base to support the incremental learning. This section explains how our detection model performs the threshold based case-base updating approach for the incremental learning in the case-base updating process on the new dataset.

4.1. Experimental Setup

To verify the ability of handling the concept drift, our proposed model is evaluated with an experiment using the new unseen dataset containing 43 features that are extracted from 32,342 Android apps samples. This experiment was conducted on a prototype system which is developed with Java SE Development Kit 8, Eclipse Integrated Development Environment Version 4.7.3a (Oxygen) with Google Core Library, Guava, Simmetrics Core Library 4.1.1, and WEKA 3.8.1 package running on a Laptop computer with Windows 8.1 64-bit operating system on core i7 processor with 8 GB RAM.

For the adaptive classifier evaluation in our prior work [4], three combination rules of voting are performed with six classification algorithms including voting with average probability, voting with the maximum probability, and the majority voting methods. Nine classification techniques were used for this experiment. In this paper, three more classification techniques are added to the existing nine classification techniques including a stacking with a logistic regression as the meta-classifier, a Neural network with two different hidden layers selection methods. Totally 12 classification techniques are executed in this experiment. The maximum detection accuracy from the 12 classification results is chosen for the final decision. Two parts of experiments are arranged to demonstrate the incremental learning ability of our proposed model on the incomplete or missing features, and the new set of features to the case-base. The detail information of the case-base editing process is explained in the following sections [24].

4.2. Dataset

An experiment is performed on our proposed model using five testing features as shown in Table 1 to verify the ability of the incremental learning for handling the concept drift. There are two categories on this testing data. First, four testing feature groups contain test feature 1–4 which were collected from the previously used dataset in our prior work [4]. This first category is collected with the aim of testing the incomplete or missing features in the case-base.

Second category contains the feature group test 5 which consists of 43 features extracted from over 24,000 malicious apps and 7535 benign apps, from 32,342 Android unique applications samples. These Android samples were gathered from the Virus Total Malware Intelligence Service [25], and Google Play. The extracted features are collected from Amos et al. [26]. This second category is collected with the aim of testing the new features to the case-base.

4.3. Adding New Cases to the Case-base

In this work, 15 new cases as shown in Table 2 are added to the case-base of our proposed system. These 15 new cases are the reduced feature set. In our previous work [4], the proposed system was

Table 1 | Profiles of data in the evaluation test sets

Dataset	Test 1	Test 2	Test 3	Test 4	Test 5
Number of features	89	112	466	49	43
Number of Phish apps	6827	10,319	7780	5000	24,807
Number of Benign apps	876	1269	1104	5000	7535
Total apps	7703	11,588	8884	10,000	32,342

evaluated on the complete feature set for the first 10 cases. In this work, the reduced feature sets, selected by the feature selection process, are used instead. Furthermore, the same five random feature combinations used in the previous work [4] are also used here as the last five cases. These five random feature combinations are created to mimic the real-world apps and defined as cases. These five combinations are randomly extracted from the 10 feature sets that form the list of frequently used features by existing anti-phishing solutions.

As same as the experimental setting of the previous work, the input features must pass through different machine learning classifiers for the case defining process and the case adding process. There are nine classifiers used in this initial step. Only nine out of 12 classifiers are used for the case defining and the case adding processes because the additional three classifiers will be used for demonstrating the benefit of the incremental learning model proposed in this work. The nine classifiers consist of six individual classifiers and three ensembles of classifiers. The six individual classifiers include C4.5 (J48), Decision Table (DT), *k*-Nearest Neighbors (IBK), Logistic Regression (LR), Naïve Bayes (NB), and SVM. Each of the three ensembles of the six classifiers used a different voting technique to produce its answer including the average of probabilities (AVG), the majority voting (MAJ), and the maximum probability (MAX) or winner-take-all. The results of all nine classifiers on all fifteen cases are shown in Table 3.

Table 2 | Profiles of all 15 cases in the proposed system

Case ID	Set of feature(s)	Number of reduced features
1	Android components	62
2	API counts	22
3	API usage actions	67
4	Security Sensitive Flows	53
5	Hardware components	5
6	Intent_action	57
7	Permission	190
8	Shell_command_strings	19
9	Contents_visual	46
10	URLs	31
11	API count + API usage + Hardware	100
12	API count + Intent	120
13	API count + API usage + Intent + Hardware	185
14	Flow + Intent	265
15	Flow + Intent + API usage + Hardware	250

The bolder italicized values shown in Table 3 represent the maximum detection accuracy among nine classifiers for each case. Then, the input features, the classifiers with their parameters, the activation function, and the final result are stored in the case-base (knowledge base) as a new case. According to the detection results shown in Table 3, the detection accuracy of each classification technique is heavily depends on the features. Similar with the experimental results shown in our previous work [4], ensembles of classifiers perform well on many cases.

4.4. Similarity Scores of Testing Data to Case-base

The proposed adaptive phishing detection system will choose the suitable classifier based on the similarity scores. If the test data found the matching case with a high similarity score (the larger the similarity score value the better match), the corresponding classifier is assigned for that test data according to the adaptive classification process. If the matching case is not found by the system, the test data will undergo to the incremental learning section in order to get an optimal solution for that test data. Then, the features, the solution, the accuracy, and the performance of that test data are stored in the case-base as the new case which is called the case-base updating process or the system learning process.

Table 4 shows the similarity scores of all the five test cases with the currently stored cases (15 cases in total). In addition, the resulting similar cases, and the suggested classifier are also presented in Table 4 when applicable.

According to the results shown in Table 4, test case 1, 2 and 4 will be proceeded with retrieving the most similar case as the solution. The most similar cases are extracted from the case-base with the highest similarity score, and the solution for that case will be used for the detection process. That is, test case 1 and 2 will use J48 as the solution while test case 4 will use the ensemble of classifiers with an average voting technique. After performing the detection process using the suggested classifier, the detection accuracy is 95.22%, 95.93% and 97.51% for test case 1, 2 and 4 respectively.

Table 3 | Detection accuracy of all 15 cases performed by nine classifiers

Case ID	J48	DT	IBK	LR	NB	SVM	AVG	MAJ	MAX
1	93.23	88.97	***93.41***	90.16	84.67	87.95	90.73	90.70	89.92
2	***95.82***	93.02	95.64	91.30	89.20	85.23	95.53	95.59	85.46
3	95.27	91.90	***95.38***	91.96	89.02	91.22	94.23	94.22	93.57
4	92.75	91.09	93.15	87.15	87.45	83.06	***93.69***	***93.69***	83.74
5	89.00	89.06	***89.12***	89.06	89.02	89.06	89.08	89.07	89.06
6	86.85	85.71	***86.90***	84.57	83.75	85.62	86.32	86.43	85.68
7	94.30	91.92	94.65	93.95	88.54	94.14	***94.78***	94.75	94.11
8	***75.40***	71.18	74.08	70.28	68.74	70.22	72.41	71.60	70.41
9	97.20	95.79	95.51	94.50	95.77	93.91	***97.52***	***97.52***	94.32
10	96.03	93.24	***97.18***	93.99	92.98	93.80	95.48	95.38	94.88
11	***95.96***	93.10	95.62	92.54	89.42	91.66	95.37	95.37	92.83
12	***94.66***	91.53	94.16	90.15	86.44	89.19	94.19	94.12	91.30
13	96.38	92.59	95.55	95.02	90.69	92.61	***96.45***	***96.45***	94.34
14	90.52	86.36	90.24	88.70	81.55	87.86	***90.77***	90.76	88.69
15	95.46	89.44	94.50	93.98	92.28	91.56	***95.80***	95.79	92.82

Table 4 | The similarity scores of all five test cases on the current cases (knowledge) in the system

Cases	Test 1	Test 2	Test 3	Test 4	Test 5
Case 1	79.42	74.40	10	1.81	1.93
Case 2	49.71	44.32	0.98	3.04	3.25
Case 3	80.28	77.34	13.01	1.74	1.86
Case 4	1.45	1.29	10.18	1.96	2.09
Case 5	18.96	21.12	8.28	6.38	***6.81***
Case 6	1.40	1.25	23.31	1.89	2.01
Case 7	0.76	0.68	0.34	1.03	1.11
Case 8	2.43	2.16	1.06	3.27	3.49
Case 9	1.56	1.39	0.68	***96.89***	2.24
Case 10	1.90	1.69	0.83	2.56	2.73
Case 11	***90.09***	***94.49***	13.89	1.42	1.52
Case 12	23.22	26.74	29.17	1.30	1.39
Case 13	64.68	63.21	35.07	1.05	1.12
Case 14	0.65	0.58	***58.62***	0.87	0.93
Case 15	41.56	39.44	43.94	0.90	0.96
Most similar case	**Case 11**	**Case 11**	No	**Case 9**	No
Classifier	J48	J48	No	Voting (AVG)	No

Bolder italicized values represent the highest similarity score among 15 results.

Since there is no similar case presented for test case 3 and 5, both cases will be sent to the incremental learning process. Test case 3 will be sent to the incremental learning process under the low similarity score condition while test case 5 will be sent to the incremental learning process under the completely new case since the similarity score is very low. The process of handling the test case 3 and 5 is called the case revising process presented in the next section.

4.5. Editing Case-base for Comprehensiveness of Adaptive Phishing Detection

Before adding any new case to the case-base, the features of the new case must pass through the feature selection process in order to prevent the storing of ineffective features. The feature selection process used in our prior work [4] is applied in this work. Therefore, the features of test case 3 and the features of test case 5 will be sent through the feature selection process. After performing the feature selection process, each test case results in a set of reduced feature. That is, there are 160 features and 26 features for test case 3 and 5, respectively.

Next, each test case will be processed by multiple classifiers including all twelve classifiers. The twelve classifiers include six individual classifies (C4.5 (J48), DT, IBK, LR, NB, and SVM), three voting ensembles (the average, the majority voting, and the maximum (winner-take-all)), a stacking ensemble with logistic regression as meta-classifier and two ANN with different numbers of hidden layers. The first ANN is with 'a' hidden layers where 'a' is the number of attributes while the second ANN is with 'o' hidden layers where 'o' is the number of classes in the case-base.

Then, the most appropriated classifiers will be selected and added into the case-base. The detection results of testing the reduced feature sets of test case 3 and 5 are shown in Table 5. According to the results presented in Table 5, the NB will be selected for test case 3 while the ANN option-a will be selected for test case 5.

Table 5 | The detection results of testing the reduced feature sets of test case 3 and 5

Classifier No		Test case 3	Test case 5
	Number of features	160	26
	Number of instances	8884	32,342
1	J48	96.60%	98.88%
2	DT	99.38%	98.35%
3	IBK	96.68%	98.54%
4	LR	98.73%	98.84%
5	NB	***99.77%***	98.42%
6	SVM	99.55%	98.83%
7	Voting-AVG	99.66%	99.15%
8	Voting-MAJ	99.66%	99.15%
9	Voting-MAX	99.30%	98.85%
10	Stacking	98.20%	98.89%
11	ANN option-a	87.97%	***99.77%***
12	ANN option-o	87.72%	99.50%
Take classifier with maximum accuracy		NB	ANN option-a

Bolder italicized values represent the highest detection accuracy among 12 classification results.

Table 6 | Profiles of four test cases

Dataset	Test case 6	Test case 7	Test case 8	Test case 9
Features number	466	43	35	30
Phish apps	343	3101	660	660
Benign apps	456	940	493	493
Total apps	799	4041	1153	1153

After the case revising process, the new feature set and its corresponding classifiers are stored as a new case in the case-base. Thus, the number of cases in the case-base is now increased from 15 to 17 cases. That is, the information of the test case 3 is stored as case 16 in the case-base while the information of the test case 5 is stored as case 17 in the case-base.

4.6. Evaluating the Newly Added Cases

As the main challenge faced by using the CBR in the phishing detection work is bringing up to date the case-base to handle the concept drift of mobile phishing. This section, the newly added case 16 and 17 will be evaluated by using 4 test cases intentionally crafting as a mismatch feature of case 16 and 17 in the case-base. Table 6 shows the profiles of the four test cases namely test case 6–9. Test case 6 testing feature groups are newly collected as the incomplete feature patterns of case 16 in the case-base whereas test case 7–9 testing feature groups are newly collected as the incomplete feature patterns of case 17 in the case-base.

The similarity scores of the four test cases on the current system with 17 cases in the case-base are presented in Table 7. While, Table 8 shows the selected case from our system to produce the answer to each test case. As can be seen that the current system can detect the correct similar case from the case-base for each test case. According to the results shown in Table 8, the accuracy of the selected classifier from the case-base is 89.34%, 99.94%, 97.32% and 97.35% for test case 6–9, respectively. While the runtime for most test cases is lower than 4 s except the runtime for test case 7 which takes almost 21 s because test case 7 contains 4041 instances.

Table 7 | The similarity scores of the four test cases on the current system with 17 cases in the case-base

Case in the case-base	Similarity score (%)			
	Test case 6	Test case 7	Test case 8	Test case 9
Case 1	10	1.93	2.14	2.31
Case 2	0.98	3.25	3.60	3.89
Case 3	13.01	1.86	2.06	2.23
Case 4	10.18	2.09	2.32	2.50
Case 5	8.28	6.81	7.55	8.16
Case 6	23.31	2.01	2.23	2.41
Case 7	0.34	1.10	1.22	1.32
Case 8	1.06	3.49	3.87	4.18
Case 9	0.68	2.24	2.49	2.69
Case 10	0.83	2.73	3.03	3.27
Case 11	13.89	1.52	1.69	1.82
Case 12	29.17	1.39	1.54	1.67
Case 13	35.07	1.12	1.24	1.34
Case 14	58.62	0.93	1.03	1.12
Case 15	43.94	0.96	1.06	1.15
Case 16	***65.51***	1.07	1.19	1.29
Case 17	0.91	***77.76***	***76.24***	***82.35***

Bolder italicized values represent the highest similarity score among 17 results.

Table 8 | The performance of the system on the four test cases

	Test case 6	Test case 7	Test case 8	Test case 9
Most similar case	16	17	17	17
Classifier	Naïve Bayes	ANN	ANN	ANN
Accuracy	89.34	99.94	97.32	97.35
Run time (s)	0.06	20.46	3.69	2.55

4.7. Confusion Matrix

In order to measure the usefulness of our proposed incremental learning model, the confusion matrix evaluation is applied. The amount of correct phishing apps prediction is represented by the True Positive (TP); the amount of incorrect phishing apps prediction is represented by the False Positive (FP); the amount of correct benign apps prediction is represented as the True Negative (TN); and the amount of incorrect benign apps prediction is represented as the False Negative (FN). These four represented outcomes are designed the confusion matrix as shown in Table 9.

The effectiveness of our adaptive phishing detection is represented with three performance measurements including accuracy expressed by Equation (3), precision expressed by Equation (4) and sensitivity expressed by Equation (5). The sensitivity expresses the ability of the proposed system to find all relevant instances in the dataset. The precision expresses the proportion of the instances that our model predicts as positive and they are actually positive. The following formulas represent their definitions, Equations (3)–(5).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

Table 9 | Confusion matrix

	Predicted Phish	Predicted benign
Actual Phish	True positive (TP)	False negative (FN)
Actual benign	False positive (FP)	True negative (TN)

Table 10 | Performance measures of adaptive phishing detection system

	Classifier	Accuracy (%)	Precision (%)	Sensitivity (%)
Case 1	IBK	93.41	84	77
Case 2	J48	95.82	97	97
Case 3	IBK	95.38	81	75
Case 4	AVG, MAJ	93.69	91	99
Case 5	IBK	89.12	59	3
Case 6	IBK	86.90	76	51
Case 7	AVG	94.78	98	92
Case 8	J48	75.40	79	70
Case 9	AVG, MAJ	97.52	98	98
Case 10	IBK	97.18	97	96
Case 11	J48	95.96	83	79
Case 12	J48	94.66	87	86
Case 13	AVG, MAJ	96.45	92	75
Case 14	AVG	90.77	84	62
Case 15	AVG	95.80	90	74
Case 16	Naïve Bayes	99.77	83	87
Case 17	ANN	99.77	99	99

The evaluation of effectiveness on our proposed model by means of accuracy, precision and sensitivity is described in Table 10. According to the results shown in Table 10, our adaptive model with incremental learning achieves a good detection accuracy for the mobile phishing. That is, most cases (14 out of 17), the accuracy is more than 90%. While, the accuracy of two cases (case 5 and 6) is more than 85% and the accuracy of case 8 is 75.4%. Also, most cases (14 out of 17), the precision is more than 80%. The precision of two cases (case 6 and 08) is more than 75% and the precision of case 5 is 59%. The sensitivity of most cases (14 out of 17) is more than 70%. The sensitivity of case 14 is 62%. The sensitivity of case 6 is 51% and the sensitivity of case 5 is 3%.

According to the results shown in Table 10, it can be seen that our adaptive model with incremental learning achieves a good detection accuracy for all cases. The proposed system can adapt to an incoming of newly and missing mobile phishing features. That is an ability of our proposed case-based reasoning system with incremental learning model to resist the concept drift in the mobile phishing context.

5. DISCUSSION AND CONCLUSION

In our prior work [3,4], the important features of the mobile phishing are identified and the adaptive mobile phishing detection model using a CBR approach is proposed to perform on a variety of features. The conducted experimental results in Kyaw Zaw and Vasupongayya [4] confirmed that the proposed adaptive detection model using CBR approach is suitable to stably handle the detection accuracy on variety of mobile phishing features that is resist to concept drift. Although the performances of the concept drift handling with the design decision of our proposed model is verified, the importance of learning process that brings the up-to-date manner to the case-base is still lacking in our previous works.

In this work, the incremental learning model for the adaptation of new cases to the case-base is proposed. The proposed incremental learning approach of the case-base updating process which is the main feature to handle the concept drift in a situation when the mismatching input features with the existing cases in the case-base occurs, is explained in detail in this work. Some experiments are performed in this work to demonstrate the adaptation performance of adding a new case to the case-base using various different testing feature groups. The proposed case-based model is evaluated using a new real-world dataset containing new strange 43 feature extracted from 32,342 Android apps. This test case is utilized in order to maintain the detection accuracy while adapting the new cases to the case-base. Using a large number of testing feature groups will satisfy the wider coverage of mobile phishing features in the real world.

The evaluation results showed that our adaptive phishing detection model well maintains the detection accuracy with stability, in the condition of the variation in the number of features occurs, and the incremental learning of case-base updating process is then performed. This work clearly show the ability of our proposed model which is effectively detecting the mobile phishing under the feature with an independent condition, since our proposed detection model is sustainable meaning it must learned and adapt to the newly phishing features in the real world by incremental manner.

5.1. Scope of the Work

The scope of the proposed adaptive mobile phishing detection model that worked on a variety of features using a CBR technique summarizes as follows:

(i) This work investigates the phishing detection on Android platform.

(ii) The proposed adaptive mobile phishing detection model initially developed and verified. The detection performance on variety of features patterns with 1065 features from 10 feature groups which are collected from 10,000 Android apps are presented. Nine classifiers are used in our previous work including six classification algorithms, and three ensembles of six base classifiers.

(iii) The incremental learning ability of the proposed adaptive phishing detection model is evaluated with a total number of 759 features which are collected from 32,342 Android apps. In addition to the previously used nine classifiers, another one ensemble technique (stacking) and two types of neural network with different numbers of hidden layers are also used in this experiment. Totally, 12 classifiers are used in this paper for the coverage of different classification techniques.

(iv) The evaluation of incremental learning model is performed with a total of nine testing feature groups for two real world circumstances containing the new strange features to the case-base and the incomplete or missing features cases.

(v) The experiment was conducted by running Weka 3.8 and Eclipse Java Oxygen 4.7.3a with Guava-package_19, simmetrics-core package_4.1.1, WekaMod and commons-codec-1.10 on a Laptop computer with core i7 processor, 8 GB RAM, and Windows 8.1 64-bit operating system.

(vi) Information gain feature selection is applied on the dataset to reduce the size of the features for overcoming the efficiency degradation.

5.2. Future Work

This work performed the continuously case-base updating with new cases that are not matched with the stored cases of our proposed detection system and reused these cases as the training examples to the case-base. The implementation of the periodic retraining of the classifier to adapt the new cases with a strange features will be addressed as the future work of this research for the sustainable learned case-base. That implementation will contribute to become more predictive of the most appropriate phishing detection algorithm.

CONFLICTS OF INTEREST

The authors declare they have no conflicts of interest.

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. This work is the extended version of previous work which is published in Journal of Computer Network and Communication, an open access journal.

REFERENCES

[1] Number of smartphone users worldwide 2014-2020, Statista, available from: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (accessed June 17, 2019) [Online].

[2] Spam and phishing in Q1, 2019 [Online], available from: https://securelist.com/spam-and-phishing-in-q1-2019/90795/ (accessed June 17, 2019).

[3] S. Kyaw Zaw, S. Vasupongayya, Revealing the important features of mobile phishing, The 13th International Conference on Knowledge, Information and Creativity Support Systems (KICSS), Pattaya, Artificial Intelligence Association of Thailand (AIAT), Thailand, 2018, pp. 222–226 [Online], available from: https://saki.siit.tu.ac.th/kicss2018/uploads_final/160__37fee-e28cdb6f56d1b4eaa673686ce50/PID5611845.pdf.

[4] S. Kyaw Zaw, S. Vasupongayya, A case-based reasoning approach for automatic adaptation of classifiers in mobile phishing detection, J. Comput. Netw. Commun. 2019 [Online], available from: https://www.hindawi.com/journals/jcnc/2019/7198435/cta/ (accessed June 21, 2019).

[5] E. Alpaydin, Introduction to machine learning, MIT Press, Cambridge, Massachusetts, 2009.

[6] S. Abu-Nimeh, D. Nappa, X. Wang, S.D. Nair, A comparison of machine learning techniques for phishing detection, Proceedings of the anti-phishing working groups 2nd annual eCrime Researchers Summit, 2007, pp. 60–69.

[7] R.S. Rao, A.R. Pais, Detection of phishing websites using an efficient feature-based machine learning framework, Neural Comput. Appl. 31 (2018), 3851–3873.

[8] R. Basnet, S. Mukkamala, A.H. Sung, Detection of phishing attacks: a machine learning approach, in: B. Prasad, Soft computing applications in industry, Springer, Berlin, Heidelberg, 2008, pp. 373–383.

[9] A.K. Jain, B.B. Gupta, Comparative analysis of features based machine learning approaches for phishing detection, 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), IEEE, New Delhi, India, 2016, pp. 2125–2130.

[10] N. Milosevic, A. Dehghantanha, K.K.R. Choo, Machine learning aided Android malware classification, Comput. Electr. Eng. 61 (2017), 266–274.

[11] T. Chin, K. Xiong, C. Hu, Phishlimiter: a phishing detection and mitigation approach using software-defined networking, IEEE Access 6 (2018), 42516–42531.

[12] P. Yi, Y. Guan, F. Zou, Y. Yao, W. Wang, T. Zhu, Web phishing detection using a deep learning framework, Wireless Commun. Mobile Comput. 2018 (2018).

[13] Baunfire.com and SparkCMS, APWG Phishing Attack Trends Report - 4Q 2018, Anti-Phishing Working Group, 2019 [Online], available from: https://www.antiphishing.org/resources/apwg-reports/ (accessed 22 March 2019) [Online].

[14] F. Toolan, J. Carthy, Phishing detection using classifier ensembles, 2009 eCrime Researchers Summit, IEEE, Tacoma, WA, USA, 2009, pp. 1–9.

[15] H.S. Hota, A.K. Shrivas, R. Hota, An ensemble model for detecting phishing attack with proposed remove-replace feature selection technique, Proc. Comput. Sci. 132 (2018), 900–907.

[16] A comparative study of phishing websites classification based on classifier ensembles, ResearchGate, Berlin, Germany, available from: https://www.researchgate.net/publication/325483941_A_Comparative_Study_of_Phishing_Websites_Classification_Based_on_Classifier_Ensembles (accessed 15 February 2019) [Online].

[17] H. Abutair, A. Belghith, S. AlAhmadi, CBR-PDS: a case-based reasoning phishing detection system, J. Ambient Intell. Humaniz. Comput. 10 (2018), 2593–2606.

[18] C. Riesbeck, R. Schank, "Inside case-based reasoning, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1989, p. 448.

[19] A. Aamodt, E. Plaza, Case-based reasoning: foundational issues, methodological variations, and system approaches, AI Commun. 7 (1994), 39–59.

[20] G. Widmer, M. Kubat, Effective learning in dynamic environments by explicit context tracking, Machine Learning: ECML-93, Springer, Berlin, Heidelberg, 1993, pp. 227–243.

[21] S. Craw, N. Wiratunga, R.C. Rowe, Learning adaptation knowledge to improve case-based reasoning, Artif. Intell. 170 (2006), 1175–1192.

[22] S.J. Delany, P. Cunningham, L. Coyle, An assessment of case-based reasoning for spam filtering, Artif. Intell. Rev. 24 (2005), 359–378.

[23] J.L. Kolodner, An introduction to case-based reasoning, Artif. Intell. Rev. 6 (1992), 3–34.

[24] J. Han, Data mining: concepts and techniques, third ed., Morgan Kaufmann Publishers, Waltham, USA, 2012, available from: https://www.oreilly.com/library/view/data-mining-concepts/9780123814791/ (accessed 08 June 2019) [Online].

[25] VirusTotal, available from: https://www.virustotal.com/gui/intelligence-overview (accessed 17 June 2019) [Online].

[26] B. Amos, H. Turner, J. White, Applying machine learning classifiers to dynamic Android malware detection at scale, 2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC), IEEE, Sardinia, Italy, 2013.