Consensus and Majority Vote Feature Selection Methods and a Detection Technique for Web Phishing

Article *in* Journal of Ambient Intelligence and Humanized Computing · January 2021

DOI: 10.1007/s12652-020-02054-3

Journal of Ambient Intelligence and Humanized Computing manuscript No.
(will be inserted by the editor)

Consensus and Majority Vote Feature Selection Methods and a Detection Technique for Web Phishing

Bandar Alotaibi · **Munif Alotaibi**

Received: date / Accepted: date

Abstract Phishing is one of the most frequently occurring forms of cybercrime that Internet users face and represents a violation of cybersecurity principles. Phishing is a fraudulent attack that is performed over the Internet with the purpose of obtaining and using without authorization the sensitive information of Internet users, such as usernames, passwords, credit card details, and bank account information. Some widely used phishing attempts involve using email spoofing or instant messaging, aiming to convince a victim to visit the spoofed websites, which will result in obtaining the victim's information. In this work, we identify and analyze the most important features needed to detect the spoofed websites in virtue of two new feature selection techniques. The first proposed feature selection technique uses underlying feature selection methods that vote on each feature, and if such methods agree on a specific feature, that feature is selected. The second feature selection technique also uses underlying feature selection methods that vote on each feature, and if the majority vote on a specific feature, the feature is selected. We also propose a phishing detection technique based on both AdaBoost and LightGBM ensemble methods to detect the spoofed websites. The proposed method achieves a very high accuracy compared to that of the existing methods.

Keywords malicious website · cybercrimes · phishing attack · phishing detection · machine learning · feature selection · cyber security

Bandar Alotaibi
University of Tabuk, Tabuk, Saudi Arabia, 71491
E-mail: b-alotaibi@ut.edu.sa

Munif Alotaibi
Shaqra University Shaqra, Saudi Arabia E-mail: munif@su.edu.sa

1 Introduction

Pretending to be a legitimate user in order to gain sensitive information in the computer networking domain is known as phishing (Abutair et al. 2019). In 2006, attacks related to phishing cost affected individuals or organizations up to $2.8 billion, and there were approximately 2.3 million victimized people. The number of phishing attacks has increased in 2007, reaching 3.6 million victims and up to $3.2 billion in losses (McCall 2007). In addition, the number of potential phishing targets increases as computer hardware costs decrease, allowing more people to have more computers, and the number of portables (e.g., laptops, tablets, and smartphones) and Internet of Things (IoT) devices (e.g., smart TVs) increases. Attackers use many techniques to lure Internet users to believe that the phishing website is the legitimate one. All these features share one concept, a Uniform Resource Locator (URL) that directs naive users to the phishing website (Thakur and Verma 2014; Verma and Dyer 2015). Therefore, this paper detects phishing attacks using URLs for two reasons: this approach can be applied to all web phishing attempts, and it is regarded as an early detection that is performed before a URL directs naive users to the website that asks for their information.

Phishing attacks are a growing serious threat to Internet users and one of the most widely occurring forms of cyberattacks used to steal sensitive information (Lastdrager 2014; Mohammad et al. 2015b; Varshney et al. 2016; Jain and Gupta 2018). It is a very difficult issue because of having to consider the massive number of emerging sites, the time it takes to identify such malicious websites, and the variety of attack forms. Thus, there is a growing need for an automated method to protect users before they access malicious websites. In this paper, we propose an automated method of identifying and labeling malicious and harmless websites before users can access them. There have been many methods proposed in the past to solve this issue, and they can be categorized into two types: proactive and reactive methods (Bahnsen et al. 2017). Proactive methods depend on blacklists of phishing websites' URLs. If a website is included in the blacklist, the network or the browsers will block it and prevent users from accessing it (Rao and Pais 2019). However, it is very difficult to keep the blacklist up to date due to the massive number of emerging websites (Bahnsen et al. 2017). On the other hand, reactive methods operate by analyzing webpages in real time. Our method belongs in the category of reactive methods due to its effectiveness. These methods usually use classification models and machine learning to detect spoofed webpages.

In the past, several machine learning algorithms have been proposed to detect phishing web sites, such as support vector machines (L'Huillier et al. 2010), random forests (Chiew et al. 2019), gradient boosting (Marchal et al. 2016), streaming analytics (Marchal et al. 2014), random field and latent Dirichlet allocation (Ramanathan and Wechsler 2013), neural networks (Mohammad et al. 2014; Feng et al. 2018), recurrent neural networks (Bahnsen et al. 2017), logistic regression (Jain and Gupta 2019), and online incremental learning (Ma et al. 2009).

1.1 Motivation

Machine learning algorithms have been used successfully to detect phishing web sites. However, there is room to improve these algorithms; one way of doing so would be to design them to detect phishing more accurately as soon as the phishing intention occurs. This opportunity for improvement motivated us to enhance algorithmic performance by proposing two different types of feature selection methods to reduce detection time. Moreover, a combination of two powerful ensemble methods (the AdaBoost classifier and the LightGBM classifier) has been investigated for its potential to improve the performance of URL phishing detection by increasing the detection rate.

1.2 Contribution

The contributions of this paper can be summarized as follows:

- We propose a consensus feature selection method that involves voting on every feature in the dataset; if all the underlying feature selection methods agree on a given feature, that feature is selected.
- We propose a majority vote feature selection method that involves voting on every feature in the dataset; if the majority (i.e., more than a half) of the base feature selection methods choose a given feature, that feature is selected.
- We also propose an accurate technique that uses a combination of the AdaBoost classifier and the LightGBM classifier to detect web phishing attacks.
- The proposed feature selection methods (i.e., the consensus and the majority vote) reduced detection time while maintaining high detection accuracy.
- Our detection method achieves better detection accuracy compared to various widely used phishing spam detection techniques.

The remainder of the paper is organized as follows. The related studies are surveyed in section 2. Section 3 introduces our web phishing attack detection technique. The results of the proposed method are described in section 4. Section 4 also discusses the results obtained with our feature selection method and our detection technique. The paper is concluded in section 5.

2 Related Work

The recent feature selection methods and detection techniques based on machine learning algorithms are surveyed in this section. Toolan and Carthy Toolan2010Feature utilized the information gain (IG) selection method to choose the most important features among forty features related to phishing and spam detection. The authors applied their technique to three different datasets to identify the most likely representative top-10 features that could be used to detect phishing in a reasonable time; their approach ended up selecting the nine most important features. The authors also used a decision tree classifier (i.e., C5.0) to detect phishing webpages, which showed the effectiveness of using higher IG values compared to using lower values.

Khonji et al. (2013) investigated several feature selection methods of detecting email phishing, including Relief-F, correlation based feature selection (CFS), IG, and Wrapper. The experimental results showed that the Wrapper method performed the best among the tested feature selection methods, while CFS was the worst feature selection method in terms of performance. The authors also tested several machine learning algorithms to identify the best algorithm that accurately detected email phishing attacks. Among the tested algorithms were random forests, support vector machines, and the decision tree method; random forest ensemble method outperformed the others.

Basnet et al. (2012) tested two well-known feature selection methods, namely, wrapper and CFS. Greedy forward selection and a generic algorithm were used to evaluate the features extracted from the webpage and the search engine. To evaluate the effectiveness of the feature selection methods, the authors used three machine learning algorithms, namely, logistic regression, random forests, and naive Bayes.

The results show that wrapper feature selection method performed better than CFC in terms of accuracy.

Chiew et al. (2019) proposed a new feature selection method called hybrid ensemble feature selection (HEFS) and designed to select the most important features used to detect web phishing attacks. The authors divided their proposed method into two phases: they used the cumulative distribution function gradient (CDF-G) in the first phase to generate primary features, and these features were fed into the second phase represented by a data perturbation ensemble and a function perturbation ensemble to produce the second subset of features. A set of baseline features resulting from the two phases was fed into machine learning algorithms used to detect phishing. The selected features were fed into several machine learning algorithms; the best algorithm in terms of accuracy was a random forest that, when used with the baseline features, obtained an accuracy of 94.6%. The authors used two tests to validate their proposed method.

Zabihimayvan et al. (2019) used the fuzzy rough set (FRS) theory to choose the most important features that could be effectively utilized to improve the performance of phishing detection. The selected features were fed into three machine learning algorithms (namely, random forest, multilayer perceptron, and sequential minimal optimization) widely used in phishing detection. The best classifier using the selected features was random forest with a 95% f-score. The authors also chose nine universal features chosen from three different datasets to be used to identify phishing attacks. These nine features were fed to the three classifiers and yielded a 93% f-score using the random forest classifier.

Rao et al. (2020) developed an application called CatchPhish to predict website phishing attacks using the URL without visiting the website. The proposed approach utilized the following components to extract important features: full URL, host name, and term frequency-inverse document frequency (TF-IDF). The selected features were divided into words hinting at phishing and fed into the random forest ensemble method to detect phishing. The authors collected their own dataset and used two more datasets to evaluate their method's performance, which showed that their method achieved promising results.

Wang et al. (2019) proposed a website phishing detection technique using URLs that would be independent from third-party services, including DNS services and search engines. The proposed method utilized deep learning to extract meaningful features in order to feed the deep learning model and detect phishing attacks. The proposed method was the first to combine the well-known deep learning techniques, namely the Recurrent Neural Network (RNN) and the Convolutional Neural Network (CNN), to detect website phishing. The proposed method achieved promising results. However, the high accuracy of deep learning-based models comes with high detection time.

Rao et al. (2019) introduced a new website phishing detection framework called PhishDump for mobile devices. The authors believe that the current detection techniques cannot work very well with mobile devices that have low computational power and a small Random Access Memory (RAM) size. Moreover, as smartphones increase in number, phishing attacks targeting smartphones continue to increase. Thus, the authors proposed a lightweight detection mechanism to detect website phishing targeting mobile devices. This mechanism combines both the SVM and Long Short Term Memory (LSTM). The proposed method has been evaluated using two data sets and has delivered promising results. Similar to the previously proposed method (Wang et al. 2019), the detection time of Rao et al.'s proposed method is an issue, since the complexity of the base estimators are high, which leads to a high detection time.

All of the previously proposed solutions share a major issue in that choosing the number of selected features is not justified. Choosing a large number of features increases the detection time, which makes it hard to detect phishing attacks in real time, while choosing a small number of features might reduce the performance of the detection method. Thus, it is important to choose the appropriate number of features. In addition, the phishing detection performance has some room for improvement.

3 Proposed Method

We introduce our proposed method in this section, divided into two subsections: the proposed feature selection methods are presented in subsection 3.1, and the proposed phishing detection technique is introduced in subsection 3.2. A summary illustration of our proposed method is presented in Fig. 1.

3.1 Feature Selection Methods

As shown in Fig. 1, we propose two feature selection methods to reduce the features in the website phishing realm. Our proposed feature selection methods utilize base feature selection methods to best generalize the base features that most likely represent the given raw data.

3.1.1 Consensus Feature Selection Method

Each feature selection method chooses the most important features on its own. The chosen features that are agreed upon by all of the base feature selection methods are the features that our selection method chooses. Formally, this is done as follows.

Scenario 1:

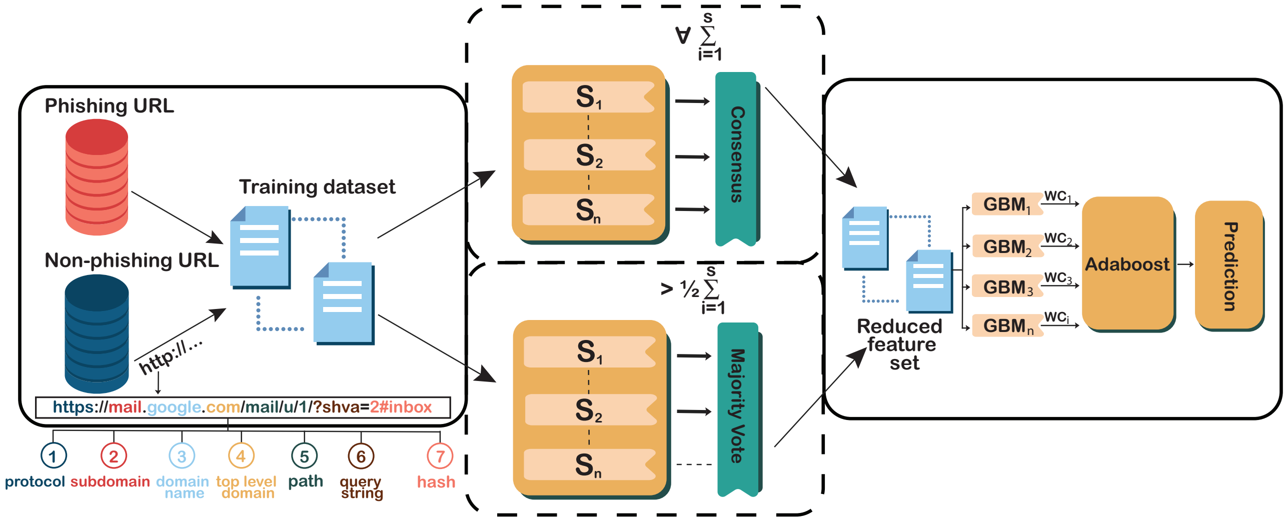


Fig. 1: Consensus and majority vote feature select methods for reducing the feature set of phishing and non-phishing URL data sources. The reduced feature set is then fed into the Adaboost-LGMB detection method

Suppose that the first base feature selection method s_1 chooses the first feature f_1, the second base feature selection method s_2 chooses the first feature f_1 as well, and the third base feature selection method s_3 chooses the first feature f_1 too. Since all the base feature selection methods agree to choose this feature (Note: a set of feature selection methods is denoted by S), f_1 is placed in the chosen feature list $\widehat{F}$, which is then represented by $\widehat{f_1}$ in our notation. Subsequently, this procedure is repeated for all the features $m = 1, 2, ..., M$.

Scenario 2:

Suppose that the first feature f_1 is chosen by the first base feature selection method s_1 and rejected by one (i.e., f_1 is rejected by s_2 or s_3) or both (i.e., f_1 is rejected by both s_2 and s_3) of the two other base feature selection methods; this feature is thus rejected. In addition, suppose that the first feature f_1 is not chosen by all the base feature selection methods s_1, s_2 and s_3; thus, this feature is rejected by our feature selection method. Thereafter, the procedure is repeated for the rest of the features $m = 1, 2, ..., M$.

The overall feature selection method is presented in Algorithm 1.

Algorithm 1 Select the Agreed-upon Features

```
Require: $F$                                   ▷ a vector of raw features
Ensure: $\hat{F}$ ▷ a feature vector selected by our feature selection method
$s_1 = Z_1(S_F)$
$s_2 = Z_2(S_F)$      ▷ $Z_n$ is a function call to a given feature selection
    method
$s_3 = Z_3(S_F)$
for $F \supset S_F$ do ▷ $S_F$ is a feature vector selected by the base feature
    selection methods
    if $F$ is selected by $\forall \sum_{i=1}^{S}$ then          ▷ ∀ means all
        $\hat{F} = F$
    else[$F$ is rejected by $\exists \sum_{i=1}^{S}$]      ▷ ∃ means at least one
        $\hat{F} \neq F$
    end if
end for
```

3.1.2 Majority Voting Feature Selection Method

Each base feature selection method chooses the most important features by itself. Subsequently, our selection method uses the majority vote on each feature in the full feature set. If a feature has been chosen by more than a half of the base feature selection methods, that feature is chosen by our feature selection method. Formally, this is done as follows.

Scenario 1:

Suppose that the first feature f_1 is chosen by the three feature selection methods (i.e., all base feature selection methods) s_1, s_2, and s_3. This feature is chosen by our feature selection method and placed in our selected feature list $\widehat{F}$. In our notation, this specific feature is denoted by $\widehat{f_1}$. All the features in the feature set $m = 1, 2, ..., M$ are processed by the same procedure.

Scenario 2:

Suppose that a specific feature (e.g., the first feature f_1) is chosen by more than a half of the base feature selection methods (i.e., it wins the majority vote). For instance, the first feature f_1 is chosen by the first base feature selection method s_1 and the second base feature selection method s_2 and rejected by the third feature selection method s_3. This feature f_1 is added to our selected feature list $\widehat{F}$ and denoted by $\widehat{f_1}$. All the features $m = 1,2,...,M$ are processed by the same procedure as well.

Scenario 3:

Suppose that the first feature f_1 is selected by one of the base feature selection methods (e.g., the first feature selection method s_1) and rejected by the two other base feature selection methods s_2 and s_3. This feature f_1 is rejected by our feature selected method. Additionally, suppose that the first feature f_1 is rejected by all the base feature selection methods s_1, s_2, and s_3. Our feature selection method rejects this feature. Thereafter, the same procedure is applied to all the features $m = 1,2,...,M$.

Our majority vote selection method is explained in more detail in Algorithm 2.

Algorithm 2 Select the Features that Have the Majority Vote

```
Require: $F$
Ensure: $\hat{F}$
$s_1 = Z_1(S_F)$
$s_2 = Z_2(S_F)$
$s_3 = Z_3(S_F)$
for $F \supset S_F$ do
    if $F$ is selected by $\forall \sum_{i=1}^{S}$ then
        $\hat{F} = F$
    else if $F$ is selected by $> \frac{1}{2} \sum_{i=1}^{S}$ then
        $\hat{F} = F$
    else[$F$ is rejected by $> \frac{1}{2} \sum_{i=1}^{S}$]
        $\hat{F} \neq F$
    end if
end for
```

3.2 Phishing Detection Technique

As shown in Fig. 1, our detection method uses a machine learning meta-algorithm (i.e., AdaBoost (Freund and Schapire 1995)) and the LGBM classifier (Ke et al. 2017). The combined algorithms are chosen because they are the most accurate algorithms we can find (in this study, more than fifteen machine learning algorithms are investigated to validate our selection methods) and because their complexity is relatively low in relation to the high performance they accomplish.

In LGBM, instead of ignoring the samples that have small gradients, the algorithm performs random sampling on them and keeps the samples that have large gradients. The samples are sorted according to their absolute values, and the samples with large gradients are chosen. The random sampling feature of this algorithm makes LGBM a perfect candidate for phishing detection because the algorithm is lightweight, and the detection thus can be performed in real time.

AdaBoost overcomes some of practical shortcomings introduced in previously proposed boosting algorithms. It uses training set $(x_1,y_1),...(x_n,y_n)$ as input, where the domain space X consists of samples or instances x_i, and the label set Y contains labels y_i. In our proposed detection technique, we aim to use binary classification where $Y = \{0,1\}$. The ultimate goal of AdaBoost is to use a base learning algorithm (i.e., a weak learner) repeatedly over several rounds $c = 1,...,C$, where C in our detection technique is the LGBM classifier. AdaBoost obtains a set of weights or a distribution over a given training data.

The distribution's weight on training sample i in each round in the base learning procedure c is denoted by $W_c(i)$. The objective of the base estimator is to discover a weak hypothesis $wh_c : X \longrightarrow \{0,1\}$ that can be applied to the distribution's weight W_c. To measure the proper weak hypothesis and calculate its error, AdaBoost applies the following equation 1:

$$\varepsilon_c = R_i \sim W_c[wh_c(x_i) \neq y_i] = \sum_{i:wh_c(x_i)\neq y_i} W_c(i) \tag{1}$$

As shown in equation 1, the error is calculated using the training samples for the base estimator used to learn with respect to the distribution's weight W_c. In our proposed detection technique, we utilize LGBM to learn from the distribution's weight W_c on the training samples. Before the first round begins, all the weights are distributed equally. Given the training set $(x_1,y_1),...(x_n,y_n)$ where $x_i \in X$ and $y_i \in Y = \{0,1\}$, the adaboost algorithm is initialized as $W_1(i) = 1/n$ and iterates through the LGBM base classifiers: $c = 1,...,C$. Each LGBM classifier is trained using the distribution's weight W_c. The weak hypothesis, $wh_c : X \longrightarrow \{0,1\}$, is achieved with error: $\varepsilon_c = R_i \sim W_c[wh_c(x_i) \neq y_i]$. As the weak hypothesis is received, a parameter, α, is chosen by adaboost (as shown in equation 2) to measure the importance of the weak hypothesis.

$$\alpha_c = \frac{1}{2} In\left(\frac{1-\varepsilon_c}{\varepsilon_c}\right) \tag{2}$$

Subsequently, as shown in equations 3 and 4, the weights are updated on the samples that are classified incorrectly and are increased to ensure that the LGBM classifier concentrates on the hard-to-learn samples in the training data.

$$W_{c+1}(i) = \frac{W_c(i)}{N_c} \times \begin{cases} e^{-\alpha_c} & \text{if } wh_c(x_i) = y_i \\ e^{\alpha_c} & \text{if } wh_c(x_i) \neq y_i \end{cases} \quad (3)$$

N_c in previous equation represents the normalization factor.

$$= \frac{W_c(i)exp(-\alpha_c y_i wh_c(x_i))}{N_c} \quad (4)$$

Eventually, as shown in equation 5, the weighted majority vote of the weak hypotheses is used to construct the final hypothesis h (Freund et al. 1999).

$$h(x) = sign\left(\sum_{c=1}^{C} \alpha_c wh_c(x)\right) \quad (5)$$

4 Results and Discussion

In this section, we present the results of our proposed selection method and detection techniques. We validate our method using two datasets; their information is shown in Table 1.

Table 1: Datasets Information

Dataset	Samples	Features	Phishing	Legitimate
Mendeley	10,000	48	5,000	5,000
Rami et al.	11,055	30	3,793	7,262

The first dataset (Tan 2018) was published in 2018 and consists of 10,000 samples, half of which are labeled as phishing websites, while the rest are labeled as legitimate websites. The number of features in this dataset is 48. The other dataset was published by Rami et al. (2015a). This dataset contains 11,055 samples, among which 3,793 are labeled as phishing web sites, while almost twice the number of samples relative to phishing websites (i.e., 7,262) are labeled as legitimate websites. The number of features is less than that of the first dataset (i.e., 30 features). We conducted our experiments using a laptop computer equipped with an Intel Core i7 CPU, 32 GB RAM and Windows 10 operating system.

4.1 Feature Selection Method Evaluation

Our feature selection method is implemented on the two datasets to select the most important features in order to reduce the detection time. In addition, both datasets were published recently, which makes them suitable options for validating our proposed method because they contain more updated samples and features of phishing websites. We use three well-known feature selection methods as base selection methods, namely, random forest, gradient boosting, and LGBM. These base feature selection methods agreed to choose 17 features in the first dataset (the Mendeley dataset), as shown in Table 2, while 9 features have been agreed upon in the second dataset (the dataset of Rami et al.). The majority of the base feature selection methods selected 23 features in the first dataset and 13 features in the second dataset.

The detection time will be reduced if we use our first feature selection method because it only chooses 35.4% of the original feature set of the Mendeley dataset. Furthermore, the detection time will decrease because our feature selection method only selects 30% of the original feature set of the Rami et al. dataset. In the next subsection, we show that our feature selection method is effective in terms of detection time while also performing well (i.e., obtaining high accuracy). Furthermore, the detection time will decrease if we use our second feature selection method because only 47.9% of the original feature set is selected by that method. In addition, the detection time will be reduced because our feature selection method reduces the original feature set by 43.3%.

4.2 Phishing Detection Technique Evaluation

We evaluate our detection technique and feature selection method using the two datasets. We follow the same data split as in (Chiew et al. 2019; Zhu et al. 2019) to be able to compare our method with that of (Chiew et al. 2019; Zhu et al. 2019) (i.e., 30% of the data are utilized for testing, while 70% of the data are reserved for training). We start with the full feature set and subsequently consider a reduced feature set (i.e., chosen by our two feature selection methods) to explore and compare the effectiveness of our feature selection methods in terms of detection time while applying our detection method. We compare our proposed selection method with that of a recent study (Chiew et al. 2019). We also compare our detection method with that of (Chiew et al. 2019) (the researchers tried several algorithms, among which the random forest ensemble method was observed to be the best algorithm in terms of accuracy), (Zhu et al. 2019), and two widely used algorithms of detecting spoofing, namely, support vector machine (SVM) and naive Bayes, to further validate our technique.

4.2.1 Evaluation Metrics

This research paper uses four statistical metrics (i.e., accuracy, precision, recall, and F1 score) to measure the effectiveness of the proposed method. These metrics rely on four indices: true negative (TN), true positive (TP), false negative (FN), and false positive (FP). Legitimate URLs are denoted as negative instances, while phishing URLs are denoted as positive instances. In this context, TN indicates the number of legitimate URL instances that are correctly detected as legitimate URLs; TP indicates the number of phishing URL samples that are accurately detected as phishing URLs; FN indicates the number of phishing URL samples that are mistakenly predicted as legitimate URLs; and FP indicates the number of legitimate URL instances that are mistakenly detected as phishing URLs.

Accuracy is an essential statistical measure of the overall effectiveness of a given model. The accuracy metric measures the number of phishing URL instances that are correctly detected (i.e., TN and TP) divided by the number of all the samples in the test set.

$$Accuracy = \frac{TN+TP}{TN+TP+FN+FP} \tag{6}$$

The precision metric measures the number of phishing URL samples that are correctly detected divided by the number of all the instances that are either correctly or incorrectly predicted to be phishing URLs.

$$Precision = \frac{TP}{FP+TP} \tag{7}$$

The recall metric measures the number of phishing URL instances that are correctly identified divided by the number of phishing URL samples that are correctly identified as phishing and the number of legitimate URL instances that are incorrectly detected as phishing.

$$Recall = \frac{TP}{FN+TP} \tag{8}$$

One of the most important statistical measures is the F1 score, which takes the weighted average of both the precision and recall metrics in order to comprehensively evaluate the performance of a given model.

$$F1\ score = 2 \times \frac{precision \times recall}{precision + recall} \tag{9}$$

Throughout this paper, all statistical measures are given a value between 0 and 100. Among the tested models, higher values indicate better performance.

Table 2: Selected features over the datasets

Mendeley (Method No. 1)	**Mendeley (Method No. 2)**	**Rami et al. (Method No. 1)**	**Rami et al. (Method No. 2)**
UrlLength	UrlLength	WebTraffic	WebTraffic
QueryLength	QueryLength	HavingSubDomain	HavingSubDomain
PctNullSelfRedirectHyperlinks	PctNullSelfRedirectHyperlinks	HavingIPAddress	HavingIPAddress
PctExtResourceUrls	PctExtResourceUrls	URLOfAnchor	URLOfAnchor
PctExtNullSelfRedirectHyperlinksRT	PctExtNullSelfRedirectHyperlinksRT	SSLfinalState	SSLfinalState
PctExtHyperlinks	PctExtHyperlinks	SFH	SFH
PathLevel	PathLevel	LinksPointingToPage	LinksPointingToPage
PathLength	PathLength	LinksInTags	LinksInTags
NumQueryComponents	NumQueryComponents	GoogleIndex	GoogleIndex
NumNumericChars	NumNumericChars		AgeOfDomain
NumDots	NumDots		RequestURL
NumDash	NumDash		PrefixSuffix
InsecureForms	InsecureForms		DNSRecord
IframeOrFrame	IframeOrFrame		
FrequentDomainNameMismatch	FrequentDomainNameMismatch		
ExtMetaScriptLinkRT	ExtMetaScriptLinkRT		
ExtFavicon	ExtFavicon		
	SubmitInfoToEmail		
	PctExtResourceUrlsRT		
	NumUnderscore		
	NumSensitiveWords		
	NumDashInHostname		
	HostnameLength		

4.2.2 Performance Measures

We evaluate and compare our detection method with existing and widely used phishing detection techniques using the accuracy measure. Figure 2 shows the accuracy of the three feature selection techniques applied to our method and two widely used spam detection algorithms, namely, support vector machines and naive Bayes. The figure shows that our method has the best accuracy on both datasets (the Mendeley dataset and the Rami et al. dataset). The figure also shows that using the feature set extracted by the proposed feature selection techniques resulted in higher accuracy and shorter computation time than those resulting from using the full feature set as-is. We perform three experiments for each dataset to further validate our two selection and detection methods. The experiments show a significant improvement in terms of accuracy on the two datasets.

For the first dataset, we compare our detection method with SVM and naive Bayes using the full feature set, the reduced feature set obtained using our first selection method (i.e., the consensus selection method), and the reduced feature set obtained using our second selection method (i.e., the majority vote selection method), as shown in Fig. 2 (a). In the first experiment, the accuracy of our detection method obtained if we use the full feature set of 48 features is higher than that of the other techniques by more than 7%. The accuracy of our method is 98.6%, and the accuracy of the method with the second-highest accuracy, namely, SVM, is 91.2%; the accuracy of the last method (naive Bayes) is 85.6%. In the second experiment, we use the reduced feature set (selected by applying our consensus feature selection method) with the three detection methods. Our detection method again is the best in terms of accuracy, and the result does not decrease considerably (i.e., only a 0.4% drop). The accuracy of our method obtained when we apply our consensus feature selection method is 98.2%. The accuracy values of the other two methods decrease as well. In the third experiment, our majority vote feature selection method is applied to improve performance and maintain an acceptable computation time (i.e., to be able to detect phishing attacks in real time). Our detection method improves slightly in terms of accuracy on the feature set extracted by applying our majority vote feature selection method; its accuracy increases by 0.3% (reaching 98.63%) in comparison to the value in the first experiment obtained when we used the full feature set. The accuracy of the two other methods decreases compared to the values observed in the first experiment.

For the second dataset, we also perform three experiments to compare our detection method with the other two detection techniques; the results are shown in Fig. 2 (b). In the first experiment, we use the full feature set of 30 features to compare the performance of the three detection methods. The accuracy of our detection method is 97.05%, which is significantly higher than that of the other two methods (the accuracy of the SVM method is 94.69%, and that of naive Bayes is 60.75%). In the second experiment, the reduced feature set (i.e., selected by our consensus feature selection method) is used. The most accurate detection method is ours, with an accuracy of 93.55%; the method with the second-highest accuracy is SVM, with an accuracy of 91.41%, and the least accurate method is naive Bayes, with an accuracy of 90.71% (representing an improvement of approximately 30% compared to the first experiment). In the third experiment, we use the feature set that has been reduced using our majority vote selection method to evaluate the three detection methods. The most accurate method again is our technique with the accuracy of 95.30%, the second method is SVM and the third method is naive Bayes.

To validate our work more precisely, we compare it with the approaches recently proposed by (Chiew et al. 2019;

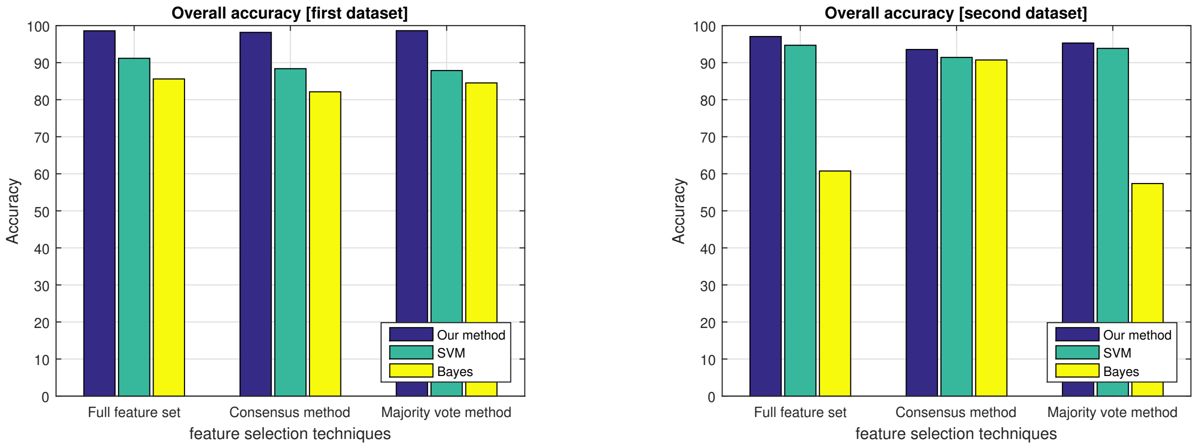


Fig. 2: Accuracy of our method compared to that of two widely used spam detection algorithms. (**a**) Accuracy of the three techniques on the Mendeley dataset. (**b**) Accuracy of the three techniques on the Rami et al. dataset.

Zhu et al. 2019) using four performance measures, namely, accuracy, precision, recall and F1 score, as shown in Table 3. Chiew et al. validated their work using two experiments on each dataset. For the first dataset, the researchers used the full feature set of 48 features in the first experiment and obtained the accuracy of 96.17%; thus, our detection method outperformed their detection method by approximately 2.5% (i.e., our method's accuracy was 98.60% when we used the full feature set). In the second experiment, the researchers applied their feature selection method that selected 10 most important features. The accuracy of their detection method on this feature set was 94.60%, which is lower than the accuracy of our detection method used with our two feature selection methods, namely, 98.17% using 17 features and 98.63% using 23 features.

For the second dataset, Chiew et al. (2019) used the full feature set of 30 features, and the accuracy of their method was 94.27%, which was also less than that of our detection method by approximately 3% (the accuracy of our detection method is 97.05%). The second experiment the researchers performed on this dataset was related to their feature selection method, in which they selected 5 features. The accuracy of the detection method when the researchers used this feature set was 93.22%, which is again less than that of our method when we use the feature sets selected by our feature selection methods, namely, 93.55% using 9 features and 95.29% using 13 features. Zhu et al. (2019) used the full feature set of 30 features, and the accuracy of their method was 96.44% (i.e., less accurate than our detection method).

4.2.3 Computation Time

The computation times of detection and training for each of the three feature selection techniques applied to our method and to the two widely used spam detection algorithms for both datasets are shown in Fig. 3. The computation times of detection for our method, SVM and naive Bayes using the full feature set are 14, 555 and 4.9 *ms*, respectively, on the first dataset, and 214, 182, and 2 *ms*, respectively, on the second dataset. The detection time of our method on the first dataset is shorter than that of the SVMs but longer than that of naive Bayes. On the other hand, the detection times of the two other methods are shorter than that of our method on the second dataset. Nonetheless, the detection times of all the methods including our method are not particularly long; thus, detection can be performed in real time.

When we use the reduced feature set extracted by the majority vote selection method, the computation time of detection for our method, SVMs and naive Bayes are 13.9, 419, and 1.9 *ms*, respectively, on the first dataset and 185, 99, and 0.97 *ms*, respectively, on the second dataset. The detection time of our method is shorter than that of SVMs and longer than that of naive Bayes when the methods are applied to the first dataset. The detection time of our method is longer than those of the other detection methods when we use the second dataset. In general, when we use the majority vote selection method, the detection times of all the methods decrease compared to when using the full feature set.

When we use the reduced feature set extracted by the consensus selection method, the computation times of detection for our method, SVMs and naive Bayes are 13.9, 363, and 1 *ms*, respectively, on the first dataset and 300, 79, and 2 *ms*, respectively, on the second dataset. Clearly, the detection times of all the methods have declined due to the reduction in the number of features (i.e., eliminating approximately two-thirds of the original features). The detection time of our detection method is shorter than that of the SVMs and longer than that of the naive Bayes when the methods are applied to the first dataset. On the other hand, the detection time of our method is longer than that of the other two methods on the second dataset.

The training times of all the detection techniques on the two datasets are not particularly long, as shown in Fig. 3. In phishing attack detection, the detection time is more impor-

Table 3: Accuracy, precision, recall, and F1 score of our method compared to that of a recent studies (Chiew et al. 2019; Zhu et al. 2019). Note: NR stands for not reported.

Method	Dataset	Feature set	Features	Accuracy	Precision	Recall	F1 score
(Chiew et al. 2019)	dataset 1	Full	48	96.17%	NR	NR	NR
(Chiew et al. 2019)	dataset 1	Baseline	10	94.60%	NR	NR	NR
Ours	dataset 1	Full	48	98.60%	98.37%	98.77%	98.57%
Ours	dataset 1	Consensus	17	98.17%	97.90%	98.37%	98.13%
Ours	dataset 1	Voting	23	**98.63%**	98.48%	98.81%	98.64%
(Chiew et al. 2019)	dataset 2	Full	30	94.27%	NR	NR	NR
(Zhu et al. 2019)	dataset 2	Full	30	96.44%	94.78%	99.02%	96.85%
Ours	dataset 2	Full	30	**97.05%**	97.09%	97.62%	97.35%
(Chiew et al. 2019)	dataset 2	Baseline	5	93.22%	NR	NR	NR
Ours	dataset 2	Consensus	9	93.55%	92.65%	95.82%	94.21%
Ours	dataset 2	Voting	13	95.29%	95.23%	96.63%	95.92%

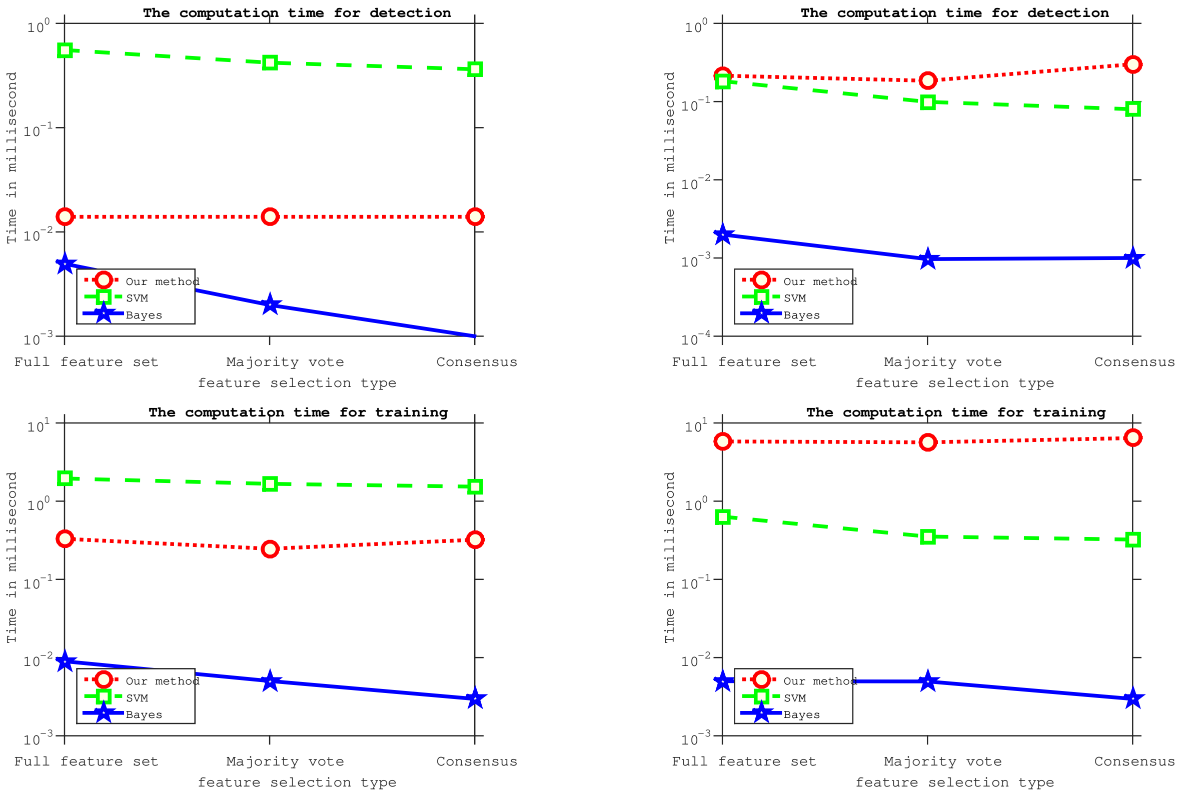


Fig. 3: Computation times of detection and training for our method compared to those of two widely used spam detection algorithms. (**a**) Computation time of detection for the three techniques on the Mendeley dataset. (**b**) Computation time of detection for the three techniques on the Rami et al. dataset. (**c**) Computation time of training for the three techniques on the Mendeley dataset. (**d**) Computation time of training for the three techniques on the Rami et al. dataset.

tant than the training time, since training is only performed once to train the detection technique.

4.3 Discussion and Limitation

Our proposed method, which uses feature selection methods and a combination of the AdaBoost classifier and the LightGBM, is more accurate than other existing techniques. Experimental results showed that our proposed method outperforms the existing methods when applied to two different popular datasets. The computation time proves that our method is suitable for real time. The detection time for one instance is very low. It can be calculated as follows:

To measure the complexity of the proposed method (the detection time for one instance is employed), the first dataset (i.e., the number of samples in this dataset, or 10,000) is utilized to examine the detection time for one instance. In the evaluation phase, 30% of the samples are assigned for testing (i.e., 3,000 samples). The total detection time of our method for the entire testing feature set (i.e., 3,000 samples) is 13.9 *ms*, as shown in Fig. 3(a).The detection time for one instance is the total detection time of the testing feature set (i.e., 13.9 *ms*) divided by the number of samples in the testing feature set (i.e., 3,000). Therefore, the detection time for one instance is 4.63 μ, which indicates that the complexity of the proposed method is very low.

Furthermore, the two feature selection methods that we proposed reduced the detection time. However, there are some challenges that face all current methods, including ours, in a real environment. First, most machine learning algorithms are susceptible to adversarial attacks; we believe such attacks can trick the machine learning algorithm to classify the phishing website as legitimate. The second challenge is that real-world URL data is very huge and thus could contain extra features that have not been taken into account or tested with our method (Xin et al. 2018). The last challenge is that benign URLs may be stopped one day for various reasons; for example, they may expire, come under attack, or simply no longer be needed. If the attacker uses these benign URLs for the goal of phishing attacks, it will be difficult to detect them.

5 Conclusions and Future Work

Internet users encounter effective attacks known as website phishing, which might be the cause of disclosure and unauthorized use of sensitive information. The sensitive information that phishers aim to steal from naive users includes credit card details, bank account information, usernames, and passwords. In this paper, we identify and analyze the most important features needed for detecting the spoofed websites. We proposed two new feature selection methods of selecting the most useful features to detect website phishing. Furthermore, we proposed a new phishing detection technique based on using two machine learning algorithms, namely, the AdaBoost classifier and the LightGBM classifier, to detect web phishing attacks. Our phishing detection technique can perform classification of phishing sites in real time while obtaining better results than those of the existing techniques. In future work, we will extend our approach from a URL-only based technique to webpage content-based technique so that we can examine and analyze the data of webpage after being rendered and downloaded to a user's computer. We believe that combining and applying both the URL-based techniques and the webpage content-based technique will add an extra layer of protection.

References

Abutair H, Belghith A, AlAhmadi S (2019) Cbr-pds: a case-based reasoning phishing detection system. Journal of Ambient Intelligence and Humanized Computing 10(7):2593–2606

Bahnsen AC, Bohorquez EC, Villegas S, Vargas J, González FA (2017) Classifying phishing urls using recurrent neural networks. In: 2017 APWG symposium on electronic crime research (eCrime), IEEE, pp 1–8

Basnet RB, Sung AH, Liu Q (2012) Feature selection for improved phishing detection. In: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer, pp 252–261

Chiew KL, Tan CL, Wong K, Yong KS, Tiong WK (2019) A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. Information Sciences 484:153–166

Feng F, Zhou Q, Shen Z, Yang X, Han L, Wang J (2018) The application of a novel neural network in the detection of phishing websites. Journal of Ambient Intelligence and Humanized Computing pp 1–15

Freund Y, Schapire RE (1995) A desicion-theoretic generalization of on-line learning and an application to boosting. In: European conference on computational learning theory, Springer, pp 23–37

Freund Y, Schapire R, Abe N (1999) A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 14(771-780):1612

Jain AK, Gupta BB (2018) Two-level authentication approach to protect from phishing attacks in real time. Journal of Ambient Intelligence and Humanized Computing 9(6):1783–1796

Jain AK, Gupta BB (2019) A machine learning based approach for phishing detection using hyperlinks information. Journal of Ambient Intelligence and Humanized Computing 10(5):2015–2028

Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu TY (2017) Lightgbm: A highly efficient gradient boosting decision tree. In: Advances in neural information processing systems, pp 3146–3154

Khonji M, Jones A, Iraqi Y (2013) An empirical evaluation for feature selection methods in phishing email classification. International Journal of Computer Systems Science & Engineering 28(1):37–51

Lastdrager EE (2014) Achieving a consensual definition of phishing based on a systematic review of the literature. Crime Science 3(1):9

L'Huillier G, Hevia A, Weber R, Rios S (2010) Latent semantic analysis and keyword extraction for phishing classification. In: 2010 IEEE international conference on intelligence and security informatics, IEEE, pp 129–131

Ma J, Saul LK, Savage S, Voelker GM (2009) Identifying suspicious urls: an application of large-scale online learning. In: Proceedings of the 26th annual international conference on machine learning, pp 681–688

Marchal S, François J, State R, Engel T (2014) Phishstorm: Detecting phishing with streaming analytics. IEEE Transactions on Network and Service Management 11(4):458–471

Marchal S, Saari K, Singh N, Asokan N (2016) Know your phish: Novel techniques for detecting phishing sites and their targets. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), IEEE, pp 323–333

McCall T (2007) Gartner survey shows phishing attacks escalated in 2007; more than $3 billion lost to these attacks. Stephane GALLAND

Mohammad R, Thabtah FA, McCluskey T (2015a) Phishing websites dataset

Mohammad RM, Thabtah F, McCluskey L (2014) Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications 25(2):443–458

Mohammad RM, Thabtah F, McCluskey L (2015b) Tutorial and critical analysis of phishing websites methods. Computer Science Review 17:1–24

Ramanathan V, Wechsler H (2013) Phishing detection and impersonated entity discovery using conditional random field and latent dirichlet allocation. Computers & Security 34:123–139

Rao RS, Pais AR (2019) Two level filtering mechanism to detect phishing sites using lightweight visual similarity approach. Journal of Ambient Intelligence and Humanized Computing pp 1–20

Rao RS, Vaishnavi T, Pais AR (2019) Phishdump: A multimodel ensemble based technique for the detection of phishing sites in mobile devices. Pervasive and Mobile Computing 60:101084

Rao RS, Vaishnavi T, Pais AR (2020) Catchphish: detection of phishing websites by inspecting urls. Journal of Ambient Intelligence and Humanized Computing 11(2):813–825

Tan CL (2018) Phishing dataset for machine learning: Feature evaluation. Mendeley

Thakur T, Verma R (2014) Catching classical and hijack-based phishing attacks. In: International Conference on Information Systems Security, Springer, pp 318–337

Varshney G, Misra M, Atrey PK (2016) A survey and classification of web phishing detection schemes. Security and Communication Networks 9(18):6266–6284

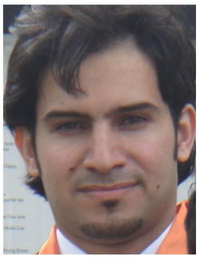
Verma R, Dyer K (2015) On the character of phishing urls: Accurate and robust statistical learning classifiers. In: Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, pp 111–122

Wang W, Zhang F, Luo X, Zhang S (2019) Pdrcnn: Precise phishing detection with recurrent convolutional neural networks. Security and Communication Networks 2019

Xin Y, Kong L, Liu Z, Chen Y, Li Y, Zhu H, Gao M, Hou H, Wang C (2018) Machine learning and deep learning methods for cybersecurity. IEEE Access 6:35365–35381

Zabihimayvan M, Doran D (2019) Fuzzy rough set feature selection to enhance phishing attack detection. In: 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, pp 1–6

Zhu E, Chen Y, Ye C, Li X, Liu F (2019) Ofs-nn: An effective phishing websites detection model based on optimal feature selection and neural network. IEEE Access 7:73271–73284



Bandar Alotaibi received the B.Sc. (Hons.) degree in computer science-information security and assurance emphasis from the University of Findlay, USA, the M.Sc. degree in information security and assurance from Robert Morris University, USA, and the Ph.D. degree in computer science and engineering from the University of Bridgeport, USA. He is an Assistant Professor with the Information Technology Department, University of Tabuk. His research interests include computer vision, network security, mobile communications, computer forensics, wireless sensor networks, and quantum computing.

Munif Alotaibi is currently an Assistant Professor in the College of Computing and Information Technology at Shaqra University. Munif received his Bachelor of Science degree in Computer Science-Information Security and Assurance Emphasis from the University of Findlay, USA, a Master of Science degree in Information Security and Assurance from Robert Morris University, USA, and a Ph.D. in Computer Science and Engineering from the University of Bridgeport, USA. His research interests include biometric authentication, pattern recognition, information security, network security, and machine learning.