Computer Systems Science & Engineering
DOI:10.32604/csse.2022.020414
Article

Tech Science Press

Phishing Websites Detection by Using Optimized Stacking Ensemble Model

Zeyad Ghaleb Al-Mekhlafi[1], Badiea Abdulkarem Mohammed[1,2,*], Mohammed Al-Sarem[3], Faisal Saeed[3], Tawfik Al-Hadhrami[4], Mohammad T. Alshammari[1], Abdulrahman Alreshidi[1] and Talal Sarheed Alshammari[1]

[1]College of Computer Science and Engineering, University of Ha'il, Ha'il, 81481, KSA
[2]College of Computer Sciences and Engineering, Hodeidah University, Hodeidah, 967, Yemen
[3]College of Computer Science and Engineering, Taibah University, Al-Madinah, 42353, KSA
[4]Nottingham Trent University, Mansfield, NG18 5BH, United Kingdom
*Corresponding Author: Badiea Abdulkarem Mohammed. Email: b.alshaibani@uoh.edu.sa
Received: 22 May 2021; Accepted: 23 June 2021

Abstract: Phishing attacks are security attacks that do not affect only individuals' or organizations' websites but may affect Internet of Things (IoT) devices and networks. IoT environment is an exposed environment for such attacks. Attackers may use thingbots software for the dispersal of hidden junk emails that are not noticed by users. Machine and deep learning and other methods were used to design detection methods for these attacks. However, there is still a need to enhance detection accuracy. Optimization of an ensemble classification method for phishing website (PW) detection is proposed in this study. A Genetic Algorithm (GA) was used for the proposed method optimization by tuning several ensemble Machine Learning (ML) methods parameters, including Random Forest (RF), AdaBoost (AB), XGBoost (XGB), Bagging (BA), GradientBoost (GB), and LightGBM (LGBM). These were accomplished by ranking the optimized classifiers to pick out the best classifiers as a base for the proposed method. A PW dataset that is made up of 4898 PWs and 6157 legitimate websites (LWs) was used for this study's experiments. As a result, detection accuracy was enhanced and reached 97.16 percent.

Keywords: Phishing websites; ensemble classifiers; optimization methods; genetic algorithm

1 Introduction

Cybercrimes became a concern of many organizations and scholars in the current years. Phishing is a type of cybercrimes that is considered one of the greatest harmful types. In phishing, the attackers stole the user's credentials and information by using false emails or websites that look like original ones. This type of attack became a concern because it affects many internet users and organizations. In phishing, a LW of a selected organization is faked by the attacker and then distributed to victims via fake or junk emails or posted URLs in social media and networks, or any medium of communication. This may lead victims to click on the links in those emails or posts which will redirect them to the fake website [1].

Internet of Things (IoT) environments are more exposed to phishing threats. In IoT, the devices are highly connected and IoT sensors can be considered as an easy attacker medium. As mentioned in [2], the smart routers, TVs, and fridges were between 25 percent of junk email hosts. Furthermore, attackers may use thingbots software of an IoT device for the dispersal of junk emails without sending any viruses or Trojans. This can be done without the user knowing, as the functionality of the IoT device may not be affected [3]. Therefore, the literature introduced several methods to increase the security of the IoT environment. However, there is no effective phishing detection method, which can effectively detect phishing emails and websites [1,4]. The literature introduced some PWs detection approaches and methods in the IoT environment. For example, a lightweight deep learning method was introduced in [5]. This method suggests the use of a detection sensor to detect PWs. The detection sensor can work in real-time and have a feature to save energy consumption. With this proposed system, IoT devices do not need to install anti-phishing software. Moreover, the detection sensor only needs to be installed once in a location between the devices and the internet local router. This method is considered somewhat efficient and can be installed in the router directly.

Several techniques were used for PW detection. Deep learning [6,7], Convolution and Deep Neural Network (CNN and DNN), Long Short-Term Memory [6], GA [7], ML [8–10], and other methods were utilized to enhance the accuracy of PW detection approaches. The results of these studies exhibited that the suggested approaches gained significant enhancement regarding sensitivity, specificity, accuracy, and other measures comparing to other modern methods. However, there is still a need to enhance the detection accuracy.

An optimized ensemble classification model for PWs detection is proposed in the present study. To build the proposed model, the GA was used. This includes three main stages that are training, ranking, and testing. First, training was used to train the classifiers (RF, AB, XGB, BA, GB, and LGBM). In this step, no optimization method was applied. In the second step, GA is used to optimize these classifiers to select the optimal values of the model parameters that can be used to increase the whole accuracy of the classifiers. Next, optimized classifiers were employed as the stacking ensemble method base classifiers based on their ranking. Finally, a test dataset is generated by collecting new websites and used to foresee the ultimate class label of the websites.

The organization of this study is as follows. An overview of the related literature is presented in Section 2. The methodology and materials were disused in Section 3. The findings of the experiments of the present study are presented in Section 4. In the same section, the findings are explained and compared with related literature. In Section 5, the conclusion of the present study is summarized in which the results and recommendations are presented.

2 Related Literature

Various research has formerly been performed in the phishing detection field. Information from related literature has been intensely reviewed to help in motivating the methodology of the present study. This related literature can be organized as follows:

2.1 IoT- Based Phishing Detection Methods

Several millions of connected IoT devices suffer from serious security issues that menace the IoT web safety [11]. Therefore, it is highly necessary to protect these IoT devices against several kinds of attacks (e.g., phishing). The targets of phishing attacks usually are unconventional networks, for instance, the IoT [1]. In [12], the main cyber menaces for the IoT industry (IIoT) have been examined and have been identified as 5 kinds of attacks, the first kind was phishing. Phishers apply compromised attacks in critical infrastructures such as IoT, an advanced approach combining zero-days malware with social

engineering and they also use other functions that are developed on remote websites to attack IIoT systems. The front-end level is used by attackers to access the IIoT.

Several methods were proposed in the IoT environment to detect PWs. Parra et al. suggested in [13] a framework that depends on cloud and deep learning comprises two tools: cloud-based temporal Long Short-Term Memory and Distributed CNN. The first tool was used for the detection of phishing as an IoT micro-security device, whereas the second tool was employed on the back-end to Botnet attacks detection and realize CNN embedding for the detection of IoT devices' distributed phishing attacks. Results from experiments demonstrated that the first tool could achieve 94.3 percent detection accuracy with CNN and 93.58 percent with F-score for phishing attacks.

Mao et al. talked in [14] about the main security concerns in intelligent IoT systems and discovered that phishing is of the most frequent types of attacks. As a proposed solution based on ML, an automated page-layout-based approach was developed by them to detect PWs. Detecting PWs in this approach is based on gaining the page layout resemblance by using aggregation analysis. In their experiments, four ML methods were employed, and the results showed improved precision.

In [15], Virat et al. he thoroughly discussed the security issues in IoT, arguing that its devices are not intelligent, making problem-solving difficult and requiring adequate methods of detection the main challenge of IoT security. Deogirikar, as well as Vidhate, have also investigated several vulnerabilities, which have endangered the IoT technology [16]. They have reviewed different IoT attacks and how to reduce their production and damage level in IoT and they have accomplished extensive research to find effective solutions.

2.2 ML-Based Detection Methods

The detection methods for several cybersecurity issues widely utilized AI and ML. Several methods based on AI and ML with good detection ability were offered for detecting PWs. Alsariera et al., for example, proposed new AI-based schemes that considered new methods of phishing mitigation [17]. Four meta-learner methods were introduced based on the extra-tree base classifiers that were applied on data sets of PW. The previous experiments' results indicate that the models achieved 97 percent accuracy and the false-positive rate was reduced to 0.028.

In the context of hyperlinks contained in HTML, Jain and Gupta in [18] proposed a new approach to detect PWs. This method brings several new hyperlink characteristics together and divides them into twelve types that are used in ML model training. The method was applied with several ML classifiers to a PW dataset. Experimental results showed that 98.4 percent accuracy was achieved with a logistic regression classifier in the proposed model. This procedure is a solution on the client-side that needs no support from third parties. Another PW detection model was introduced by Feng via a neural system [19]. This model used a Montecarlo technique during the training stage and accuracy that was achieved reached 97.71 percent, with False Positive Rate reaching 1.7 percent, signifying that the suggested model is worthy in comparison to other ML methods of PWs detection.

In order to predict PWs, Aburub and Hadi in [20] used association rules. The phishing multi-class Association Rule system was employed with a dataset that contains 10,068 legitimate and PWs, which is comparable to other associative classification methods. Their findings showed that their method gained acceptance rate of detection. Likewise, there have been other ML-based methods using selection techniques [21,22], ensemble classifiers [23], hybrid deep learning and ML methods [24], and other methods.

3 Materials and Methods

The present section presents and explains the suggested genetic-based ensemble classifier technique for enhancing PW detection. The methodology that was followed in the present study is shown in Fig. 1. Three main stages constitute the methodology of the present study: training, ranking, and testing stages. These stages are more discussed in the following sub-sections. The training stage aims to train the classifiers (RF, AB, XGB, BA, GB, and LGBM) without optimization. The purpose after that is: first, to get an overview of the ensemble classification performance before optimization, and second, to discover which of the PW characteristics are most valuable. The GA is then used to optimize the above-mentioned classifiers. Here, the GA was used to increase the whole accuracy of the suggested model by picking the optimal values of model parameters. Next, in the ranking stage, the stacking method was used to arrange the optimized classifiers and build an ensemble classifier. In the testing phase, testing data was gathered and used.

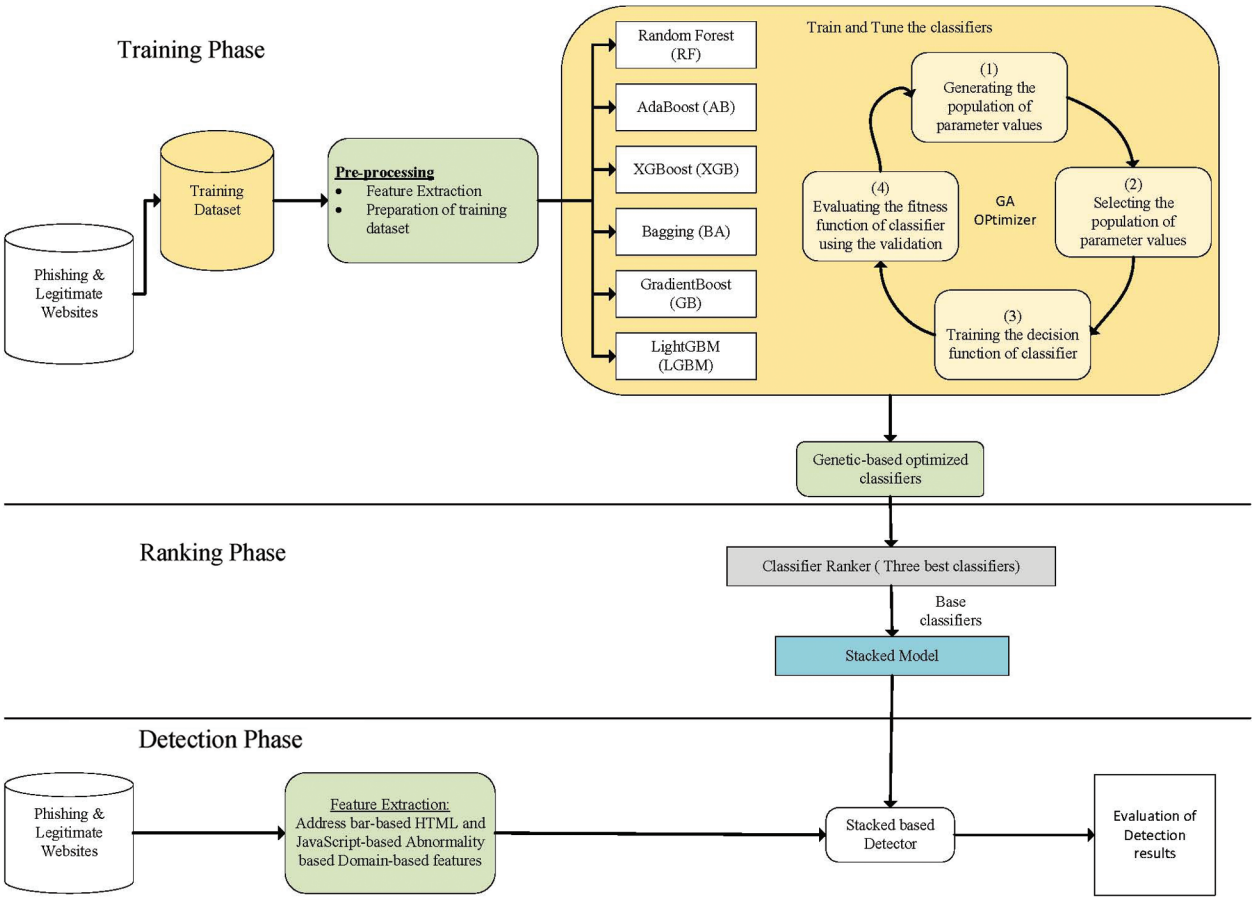

Figure 1: Proposed model for PW detection

The methodology in [25] has been followed for websites' features excerption. PhishTank database was used in this methodology. Malware and Phishing Blacklist has been gathered from a range of benign and malicious websites [26]. A Python script is created with the Whois, urllib, ipaddress, requests and Beautiful Soup libraries so that the features that were used in the dataset of training (abnormality-based features, domain-based features, bar-based features and HTML and JavaScript features) were removed.

Subsequently, these features were ultimately provided into the classifiers to forecast the website's ultimate label of the class.

Experiments in the present study were conducted on a public dataset available in the UCI ML Repository [27]. Scripts were written in the Anaconda environment under windows 10 64-bit. The Python 3.6 language was used. The employed dataset that was used in these experiments is comprised of 6157 LWs (56 percent) and 4898 PWs (44 percent). The number of minority classes was increased by the oversampling technique to imbalance the dataset. SVM-SMOTE that incorporate the SVM algorithm to identify the misclassification points, were used in the present study. The features of the dataset (30 features) can be classified into 4 groups: domain-based (7 features), abnormality-based (6 features), HTML and JavaScript-based (5 features), and address bar-based (12 features). Tab. 1 lists the names of these features, as well as the Python libraries that were used to extract each one during the testing phase of the project.

To assess the ensemble model, specific performance measures were utilized. These measures are classification accuracy, recall, precision, F-score, false-negative rate (FNR), and false-positive rate (FPR). Commonly, numerous researches used these measures to assess the PW detection systems' performance [10]. The measures were computed as shown in Eqs. (1)–(6) respectively.

- Accuracy: It is the ratio of tweets that are successfully predicted and accurately specified to the entire dataset D.

$$Acc = \frac{TP + TN}{|D|} \tag{1}$$

- Recall: It is the number of rumor tweets accurately predicted (TPs) to the total actual tweet numbers (TP+FNs).

$$R = \frac{TP}{TP + FN} \tag{2}$$

- Precision: It is the number of rumor tweets (TP) accurately predicted to total predicted rumor tweets (TP+FP).

$$P = \frac{TP}{TP + FP} \tag{3}$$

- F1-score: It is precision and recall harmonic mean. It balanced assessment between precision and recall.

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{4}$$

- False-Positive Rate (FPR): It is the total negative numbers are divided by the number of incorrect positive predictions.

$$FPR = \frac{FP}{TN + FP} \tag{5}$$

- False-Negative Rate (FNP): It is the percentage of the incorrectly classified PWs.

$$FNP = \frac{FN}{TP + FN} \tag{6}$$

Table 1: PWs features description

Feature category	Feature name	Description	Python library used
Address bar-based	having_IP_Address	Using the IP Address	IPaddress Urllib Re Datetime BeautifulSoup Socket
	URL_Length	Long URL to hide the suspicious part	
	Shortening_Service	Using shortening service	
	having_At_Symbol	URL having @ symbol	
	double_slash_redirecting	URL uses "//" symbol	
	Prefix_Suffix	Add prefix or suffix separated by (-)	
	having_Sub_Domain	Website has sub domain or multi sub domain	
	SSLfinal_State	Age of SSL certificate	
	Domain_registeration_length	Domain registration length	
	Favicon	Associated graphic image (icon) with webpage	
	Port	Open port	
	HTTPS_token	Presence of HTTP/HTTPS in domain name	
HTML and JavaScript-based	Redirect	How many times a website has been redirected	Request BeautifulSoup
	on_mouseover	Effect of mouse over on status bar	
	RightClick	Disabling right click	
	popUpWindow	Using pop-up window to submit personal information	
	Iframe	Using Iframe	
Abnormality based	Request_URL	% of external objects contained within a webpage	BeautifulSoup Re WHOIS
	URL_of_Anchor	% of URL Anchor (<a> tag)	
	Links_in_tags	% of links in <meta>, <script> and <link>	
	SFH	Server from Handler	
	Submitting_to_email	Submit user information using mail or mailto	
	Abnormal_URL	Host name in URL	
Domain-based features	age_of_domain	Age of the website	WHOIS Urllib BeautifulSoup
	DNSRecord	Website in WHOIS dataset	
	web_traffic	Popularity of the website	
	Page_Rank	Page Rank	

(Continued)

Table 1 (continued).

Feature category	Feature name	Description	Python library used
	Google_Index	Google Index	
	Links_pointing_to_page	# of links pointing to page	
	Statistical_report'	found in statistical reports	
	Result	Website is classified as phishing or legitimate	

In order to assess the suggested method accurately, 10-fold cross-validation was used with all the conducted experiments with optimized or non-optimized classifiers. The normality of each fold was also checked.

4 Results and Discussion

The present section designates each method's results before comparing them with the related works with proper discussion.

4.1 Experimental Results in Training Stage

As mentioned above, 10-fold cross-validation was used to train a set of ensemble classifiers. The experiment was first conducted without the use of GA. The default configuration classifier performance is shown in Tabs. 2–5 to obtain the highest precision compared with other classifiers. RF classification was the best and achieved approximately 97 percent accuracy. The rest of the classifiers achieved an accuracy ranging from 93 percent to 94.61 percent. BA also attained a good accuracy of 96.73 percent and LGBM of 96.53 percent.

Table 2: Accuracy of ensemble classifiers

Fold	RF (%)	AB (%)	XGB (%)	BA (%)	GB (%)	LGBM (%)
1	96.745	93.309	94.937	96.383	94.937	96.745
2	96.835	92.495	94.575	96.474	95.208	96.383
3	96.835	93.580	93.942	96.203	93.942	96.022
4	97.649	93.219	94.304	97.378	94.485	96.745
5	96.926	92.676	94.394	96.745	94.485	95.841
6	96.471	92.489	94.208	96.742	94.027	96.742
7	97.376	92.851	94.027	96.742	93.937	96.471
8	97.738	94.389	95.656	97.738	95.837	97.647
9	97.195	93.303	94.389	96.833	94.389	97.014
10	96.471	93.394	94.027	96.018	94.842	95.656
Average	97.024	93.171	94.446	96.726	94.609	96.527

Table 3: Precision of ensemble classifiers

Fold	RF (%)	AB (%)	XGB (%)	BA (%)	GB (%)	LGBM (%)
1	96.530	93.250	94.945	96.830	94.945	96.672
2	95.969	92.284	94.127	96.118	94.462	96.389
3	96.724	94.322	94.921	96.262	95.208	96.524
4	97.488	93.819	94.913	97.464	95.215	97.284
5	96.926	91.824	93.939	95.994	93.671	95.072
6	95.631	90.738	92.698	96.278	92.405	97.010
7	96.332	92.357	93.323	95.687	93.038	95.981
8	97.727	93.810	95.498	98.036	95.806	98.039
9	96.844	93.058	94.196	95.847	94.343	97.306
10	95.645	92.628	93.120	95.292	94.771	95.285
Average	96.582	94.695	94.519	94.985	94.865	95.147

Table 4: Recall of ensemble classifiers

Fold	RF (%)	AB (%)	XGB (%)	BA (%)	GB (%)	LGBM (%)
1	98.240	95.040	96.160	97.440	96.160	97.600
2	98.256	94.770	96.513	98.415	97.306	97.306
3	97.788	94.471	94.471	97.156	94.155	96.524
4	98.248	94.268	95.064	98.089	95.064	96.975
5	97.553	95.269	96.085	97.553	96.574	97.553
6	98.339	96.013	97.010	98.173	97.010	97.009
7	98.691	94.926	96.072	97.545	96.236	97.709
8	97.557	96.254	96.743	97.883	96.743	97.720
9	98.319	94.622	95.462	97.815	95.294	97.143
10	97.851	95.537	96.198	97.355	95.868	96.860
Average	98.084	96.601	96.393	96.730	96.593	96.700

Fig. 2 shows FPR and FNR. It was found that RF achieved a distinguished FPR (0.05) and FNR (0.02). LGBM achieved the second range with an FPR that equals 0,068. The GB classification obtained a 0.07 value of FPR. The lowest FNR value was also found in the RF (0.02), followed by AB and BA. while the AB classification had low FNR, FPR levels were higher than those of the LGBM, which means that a false alarm is likely to be raised. If the true value is negative, then a positive result is given.

Table 5: F-score of ensemble classifiers

Fold	RF (%)	AB (%)	XGB (%)	BA (%)	GB (%)	LGBM (%)
1	97.310	94.136	95.548	96.970	95.548	97.134
2	97.565	93.511	95.305	97.251	95.863	96.845
3	97.408	94.396	94.695	96.471	94.678	96.524
4	97.940	94.043	94.988	97.623	95.139	97.129
5	97.486	93.515	95.000	97.407	95.100	96.296
6	97.222	93.301	94.805	97.044	94.652	97.010
7	97.731	93.624	94.677	96.916	94.610	96.837
8	97.963	95.016	96.117	97.800	96.272	97.879
9	97.496	93.833	94.825	97.333	94.816	97.225
10	96.727	94.060	94.634	96.327	95.316	96.066
Average	97.485	95.714	95.496	95.900	95.760	95.949

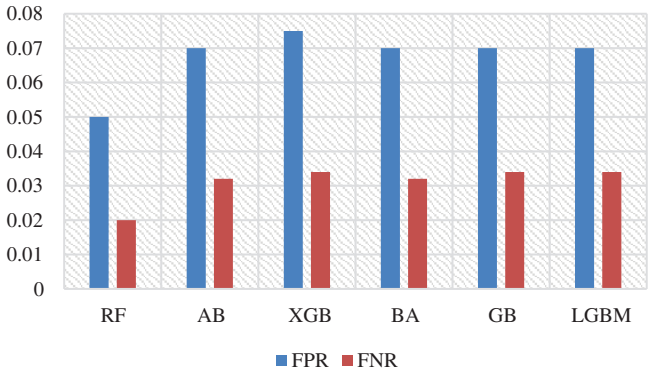

Figure 2: Ensemble methods' FPR and FNR

4.2 Experimental Results of Ranking and Testing Stages

While all classifiers have demonstrated good performance, several parameters have to be adjusted to achieve better assessment results. For each classifier, it is relatively difficult to adjust such parameters. The GA in the present study is used to adjust the parameters of classifiers. In the area of algorithm parameter search, the GA has shown good results [28]. For configuring the GA, the following parameters were used (see Tab. 6).

Table 6: GA parameter settings that used in the present study

Parameter	Value
Population size	24
Generations	10
Mutation rate	0.02
Early stop	12
Crossover rate	0.5

As a result of the multitude of configurable parameters, Tab. 7 displays the list of each classifier's adjusted parameters and the GA optimized parameters. The learning rate and optimal number of estimators obtained the most important parameters, among all parameters that have a significant impact on the classification's performance.

Table 7: Optimized parameters of classifier list

Classifier name	Adjusted parameters	Best GA-based configuration
RF	Criterion: [‘entropy’, ‘gini’] max_depth: [10–1200] + [None] max_features: [‘auto’, ‘sqrt’,'log2’, None] min_samples_leaf: [4–12] min_samples_split: [5–10] n_estimators': [150–1200]	Criterion: entropy max_depth: 142 min_samples_leaf: 4 min_samples_split: 5 n_estimators: 1200
AB	n_estimators: [100–1200] learning_rate: [1e−3, 1e−2, 1e−1, 0.5, 1.0]	learning_rate: 0.1 n_estimators: 711
XGB	n_estimators: [100–1200] max_depth: [1–11], learning_rate: [1e−3, 1e−2, 1e−1, 0.5, 1.] subsample: [0.05–1.01] min_child_weight: [1–21]	learning_rate: 0.1 max_depth: 5 min_child_weight: 3.0 n_estimators: 588 subsample: 0.7
BA	n_estimators: [100–1200] max_samples: [0.1, 0.2, 0.3, 0.4, 0.5, 1.0, 1.1] bootstrap: [True, False]	n_estimators: 1077 max_samples: 0.5 bootstrap: True
GB	n_estimators: [100–1200] learning_rate: [1e−3, 1e−2, 1e−1, 0.5, 1.0] subsample: [0.05–1.01] max_depth: [10–1200] + None min_samples_split: [5–10] min_samples_leaf: [4–12] max_features: [‘auto’, ‘sqrt’,'log2’, None]	n_estimators: 344 learning_rate: 1.0 subsample: 1.0 max_depth: 1067 min_samples_split: 5 min_samples_leaf: 12 max_features: ‘auto'
LGBM	boosting_type: [‘gbdt’, ‘dart’, ‘goss’, ‘rf’] num_leaves: [5–42] max_depth: [10–1200] + None learning_rate: [1e−3, 1e−2, 1e−1, 0.5, 1.] n_estimators: [100–1200] min_child_samples: [100, 500] min_child_weight: [1e−5, 1e−3, 1e−2, 1e−1, 1, 1e1, 1e2, 1e3, 1e4] subsample: sp_uniform(loc = 0.2, scale = 0.8) colsample_bytree': sp_uniform(loc = 0.4, scale = 0.6) reg_alpha: [0, 1e−1, 1, 2, 5, 7, 10, 50, 100], reg_lambda: [0, 1e−1, 1, 5, 10, 20, 50, 100], min_split_gain: 0.0, subsample_for_bin: 200000	boosting_type: ‘gbdt’ num_leaves: 13 max_depth: 15 learning_rate: 0.5 n_estimators: 500 min_child_samples: 399 min_child_weight: 0.1 subsample: 0.855 colsample_bytree: 0.9234 reg_alpha: 2 reg_lambda: 5 min_split_gain: 0.0, subsample_for_bin: 200000

Compared with the default parameters in Tab. 8, XGB and GB have significantly improved. In the meantime, both the LGBM and RF classification performance has been reduced.

Table 8: Optimized ensemble models accuracy

Fold	GA-RF (%)	GA-AB (%)	GA-XGB (%)	GA-BA (%)	GA-GB (%)	GA-LGBM (%)
1	97.110	94.850	96.745	96.560	97.110	96.840
2	96.840	93.130	97.016	96.750	96.930	96.470
3	97.200	93.040	96.925	96.560	96.930	95.660
4	96.020	93.760	97.468	97.650	97.830	96.200
5	96.290	92.950	97.016	97.020	97.020	96.200
6	96.470	93.570	96.923	96.740	97.010	96.200
7	96.740	92.850	97.285	97.010	97.290	96.830
8	97.830	95.660	97.556	97.470	97.830	97.470
9	97.010	92.850	97.285	97.010	97.190	96.560
10	95.930	93.670	95.927	96.200	96.110	95.750
Average	96.744	93.633	97.014	96.897	97.125	96.418

In addition, RF, XGB, Gradient Boost and LGBM confusion matrices are illustrated in Figs. 3–6.

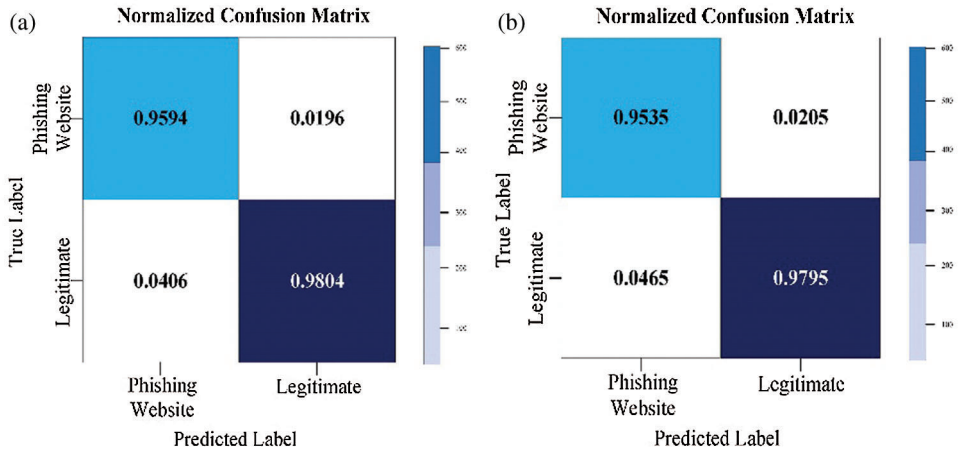

Figure 3: Normalized RF confusion matrix for the PW (a) with default parameters; (b) with optimized parameters

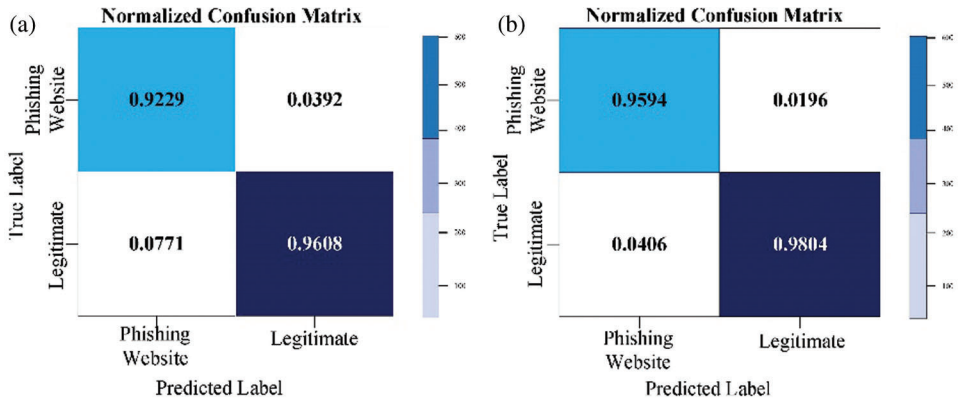

Figure 4: Normalized XGB confusion matrix for the PW (a) with default parameters; (b) with optimized parameters

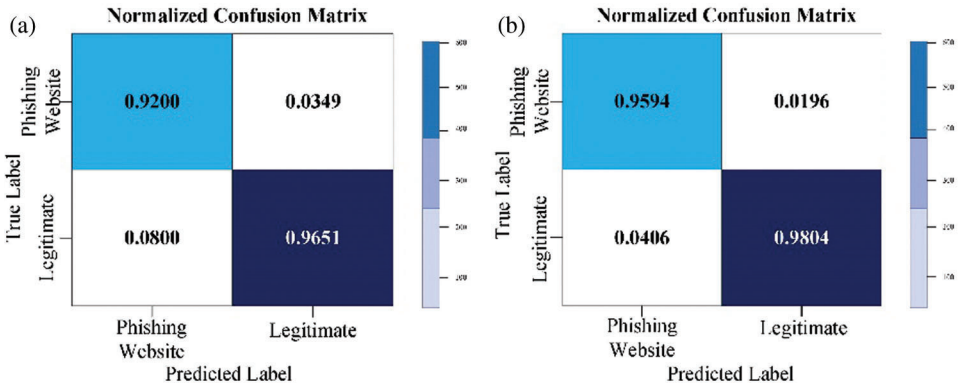

Figure 5: Normalized GB confusion matrix for the PW (a) with default parameters; (b) with optimized parameters

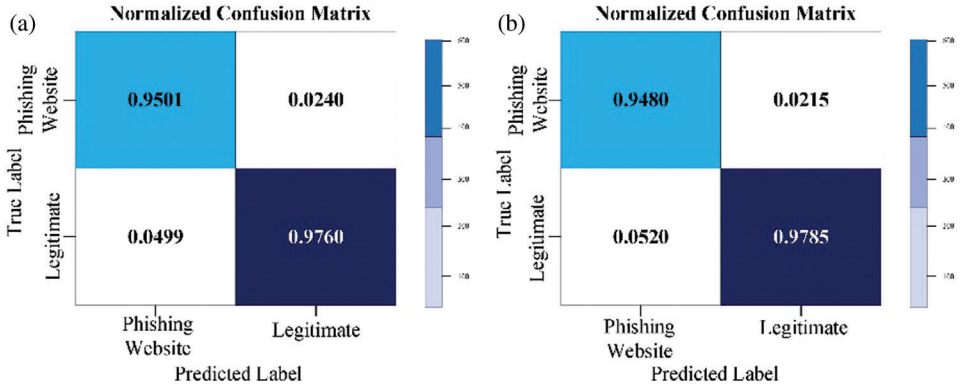

Figure 6: Normalized LGBM confusion matrix for the PW (a) with default parameters; (b) with optimized parameters

In Fig. 4b and Fig. 5b, it is noticeable that the classifier GA-XGB and GA-GB have been optimized to the greatest benefit. 95.94 percent of cases have been detected correctly as a PW class, representing the TP measure and 4.06 percent of instances incorrectly as 'legitimate' class, which is the FP measure. Furthermore, 98.04 percent of instances were detected as the "legitimate" class representing the TN measure, whereas the FN measure represented incorrectly 1.96 percent of detected instances as a PW class. It is possible to conclude that GA-XGB and GA-GB classification systems have achieved a high TP rate and a low FP. The results for other performance measures are listed in Tab. 9.

Table 9: Results for performance evaluation measures when detecting phishing and legitimate classes

Classifier	Class name	Precision	Recall	F-score
GA-RF	PW	0.964	0.941	0.951
	Legitimate	0.952	0.973	0.964
	Weighted Average	0.959	0.957	0.959
GA-XGB	PW	0.975	0.958	0.965
	Legitimate	0.967	0.980	0.972
	Weighted Average	0.970	0.970	0.970
GA-based GB	PW	0.970	0.957	0.964
	Legitimate	0.968	0.975	0.971
	Weighted Average	0.969	0.968	0.968
GA- LGBM	PW	0.951	0.942	0.947
	Legitimate	0.955	0.963	0.958
	Weighted Average	0.953	0.953	0.953

The performance of these classifiers was classified after the training phase, with the best three models being GA-GB, GA-XGB, and GA-BA. These models were used as base classifiers of a stacking ensemble method in the next stage. In the meta-learner, RF, GB, and Support Vector Machine (SVM) classification were investigated. Finally, the achieved accuracy reached 97.16 percent, which in the previous phase exceeds the other ensemble methods (see Tab. 10).

4.3 Statistical Analysis

Tab. 11 compares the results obtained by preliminary settings where classifiers are trained using standard hyperparameter settings, and by adjusting the hyperparameters in a classifier after application of GA. The mean of precision and variance values for each classifier is also summarized. The results also show that the mean of the GB classification by means of GA optimization was above the level of all other classification devices before and after optimization.

The statistical significance was also measured in several stages, besides the basic statistical measurements listed above. The two samples for the mean t-test were therefore used. This comparison zero hypothesis, h_0, is that 'precise methods of GA optimization are identical in classifiers before and after application.' The p values show that the null hypothesis can be rejected with an accepted level of significance, so there were statistically significant improvements in AB, XGB, BA, and GB (see Tab. 12).

Table 10: The accuracy of the optimized stacking ensemble method

Fold	RF level	GB	SVM
1	97.378	96.926	96.835
2	96.835	96.383	96.564
3	96.745	96.745	97.107
4	97.197	96.926	97.649
5	96.926	96.835	97.197
6	96.923	97.014	97.466
7	97.285	96.742	97.104
8	97.647	97.466	97.738
9	97.195	97.195	97.647
10	95.837	95.928	96.290
Average	96.997	96.816	97.160

Table 11: The GA-optimized classifier class average accuracy and variance values

Classifier name		Without optimization	With GA optimization
RF	Avg.	0.97024	0.96744
	Variance	2.02645E-05	3.41027E-05
AB	Avg.	0.93171	0.93633
	Variance	3.34352E-05	8.78734E-05
XGB	Avg.	0.94446	0.97007
	Variance	2.67697E-05	2.21557E-05
BA	Avg.	0.96726	0.96897
	Variance	2.68518E-05	1.88357E-05
GB	Avg.	0.94609	0.97125
	Variance	3.71454E-05	2.39583E-05
LGBM	Avg.	0.96527	0.96418
	Variance	3.49E-05	2.93418E-05

Moreover, a comparison with the earlier studies that used the same PWs as shown in Tab. 13 was made. The measurements included accuracy, precision, and recall. The results demonstrated that the proposed method overtook other related and recent projects in both [7] and [10].

Table 12: The findings of p values and $t-tests$

Classifier name		t-test result	Conclusion
RF	t-stat.	1.466706885	No significant
	p-value	0.08825352	improvement
AB	t-stat.	−2.100040666	Significant
	p-value	0.032556993	improvement
XGB	t-stat.	−13.49130461	Significant
	p-value	1.41117E-07	improvement
BA	t-stat.	−2.976672182	Significant
	p-value	0.007766628	improvement
GB	t-stat.	−11.26647694	Significant
	p-value	6.57633E-07	improvement
LGBM	t-stat.	0.971025	No significant
	p-value	0.178454	improvement

Table 13: Comparison with related studies of the proposed method

Paper	Classifier	Acc. (%)	Precision	Recall
Ali and Ahmed [7]	GA-ANN	88.77	0.8581	0.9334
Ali and Malebary [10]	POS-RF	96.83	0.9876	0.9537
The proposed model	Optimized Stacking Ensemble	97.16	0.9686	0.9683

5 Conclusion

This paper proposes, to detect phishing sites, an optimized stacking ensemble model. The optimization method has been used to identify the optimized parameter values of several ensemble learning methods by using a GA. Training, ranking, and testing are the three stages that form the proposed model. In the training stage, several ensemble learning methods have been trained, including RF, AB, XGB, BA, GB, and LGBM, without using the GA method. GA is then used to optimize these classifiers by selecting the optimum model parameter values and enhancing whole precision. In the ranking phase, certain classifiers were used as basis classifiers for the stacking ensemble method. These classifiers were the best ensemble methods (GA-GB, GA-XGB, and GA-BA). Finally, new websites were compiled in the testing phase and used as a test data set to guess the ultimate label of the class (legitimate or phishing). The experiments' findings demonstrate a higher performance compared to other ML-based detection methods with the proposed optimized stacking ensemble method. The accuracy achieved amounted to 97.16 percent. To prove that the acquired improvements were statistically significant, a statistical analysis was performed. In addition, the findings showed that the proposed methods got higher accuracy compared with recent studies that used the same phishing dataset. As a recommendation for future studies, more light detection methods will be more accurate with IoT environments. Furthermore, using deep learning methods to investigate and improve the detection rate of PWs and using more phishing datasets is also advisable.

Acknowledgement: Our acknowledgement for the funding of this research goes to the Scientific Research Deanship at the University of Ha'il, Saudi Arabia.

Funding Statement: This research has been funded by the Scientific Research Deanship at University of Ha'il-Saudi Arabia through Project Number RG-20 023.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] B. B. Gupta, N. A. Arachchilage and K. E. Psannis, "Defending against phishing attacks: Taxonomy of methods, current issues and future directions," *Telecommunication Systems*, vol. 67, no. 2, pp. 247–267, 2018.

[2] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2018.

[3] R. Roman, P. Najera and J. Lopez, "Securing the internet of things," *Computer*, vol. 44, no. 9, pp. 51–58, 2011.

[4] D. Tang, *Event Detection in Sensor Networks*, USA: School of Engineering and Applied Sciences, George Washington University, 2009.

[5] B. Wei, R. A. Hamad, L. Yang, X. He, H. Wang *et al.*, "A deep-learning-driven light-weight phishing detection sensor," *Sensors*, vol. 19, no. 19, 4258, pp. 1–13, 2019.

[6] M. Somesha, A. R. Pais, R. S. Rao and V. S. Rathour, "Efficient deep learning techniques for the detection of phishing websites," *Sādhanā*, vol. 45, no. 1, pp. 1–18, 2020.

[7] W. Ali and A. A. Ahmed, "Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting," *IET Information Security*, vol. 13, no. 6, pp. 659–669, 2019.

[8] K. L. Chiew, C. L. Tan, K. Wong, K. S. Yong and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Information Sciences*, vol. 484, pp. 153–166, 2019.

[9] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," *Neural Computing and Applications*, vol. 31, no. 8, pp. 3851–3873, 2019.

[10] W. Ali and S. Malebary, "Particle swarm optimization-based feature weighting for improving intelligent phishing website detection," *IEEE Access*, vol. 8, pp. 116766–116780, 2020.

[11] F. Khursheeed, M. Sami-Ud-Din, I. A. Sumra and M. A. Safder, "Review of security mechanism in internet of things (IoT)," in *Proc. ICACS*, Lahore, Pakistan, pp. 1–9, 2020.

[12] K. Tsiknas, D. Taketzis, K. Demertzis and C. Skianis, "Cyber threats to industrial IoT: A survey on attacks and countermeasures," *IoT*, vol. 2, no. 1, pp. 163–118, 2021.

[13] G. D. Parra, P. Rad, K. K. Choo and N. Beebe, "Detecting internet of things attacks using distributed deep learning," *Journal of Network and Computer Applications*, vol. 163, no. 102662, pp. 1–20, 2020.

[14] J. Mao, J. Bian, W. Tian, S. Zhu, T. Wei *et al.*, "Phishing page detection via learning classifiers from page layout feature," *EURASIP Journal on Wireless Communications and Networking*, vol. 1, pp. 1–14, 2019.

[15] M. S. Virat, S. M. Bindu, B. Aishwarya, B. N. Dhanush and M. R. Kounte, "*Security and privacy challenges in internet of things*," in *Proc. ICOEI*, Tirunelveli, India, pp. 454–460, 2018.

[16] J. Deogirikar and A. Vidhate, "Security attacks in IoT: A survey," in *Proc. I-SMAC*, Palladam, Indea, pp. 32–37, 2017.

[17] Y. A. Alsariera, V. E. Adeyemo, A. O. Balogun and A. K. Alazzawi, "AI Meta-learners and extra-trees algorithm for the detection of phishing websites," *IEEE Access*, vol. 8, pp. 142532–142542, 2020.

[18] A. K. Jain and B. B. Gupta, "A machine learning based approach for phishing detection using hyperlinks information," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 5, pp. 2015–2028, 2019.

[19] F. Feng, Q. Zhou, Z. Shen, X. Yang, L. Han *et al.*, "The application of a novel neural network in the detection of phishing websites," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–15, 2018.

[20] F. Aburub and W. Hadi, "A new association classification based method for detecting phishing websites," *Journal of Theoretical and Applied Information Technology*, vol. 99, no. 1, pp. 147–158, 2021.

[21] E. Gandotra and D. Gupta, "An efficient approach for phishing detection using machine learning," in *Multimedia Security*, Singapore: Springer, pp. 239–253, 2021.

[22] S. Shabudin, N. S. Sani, K. A. Ariffin and M. Aliff, "Feature selection for phishing website classification," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 4, pp. 587–595, 2020.

[23] A. Subasi, E. Molah, F. Almkallawi and T. J. Chaudhery, "Intelligent phishing website detection using random forest classifier," in *Proc. ICECTA*, Ras Al Khaimah, UAE, pp. 1–5, 2017.

[24] X. Yu, "Phishing websites detection based on hybrid model of deep belief network and support vector machine," *IOP Conference Series: Earth and Environmental Science*, vol. 602, no. 1, 12001, pp. 1–9, 2020.

[25] D. R. Patil and J. B. Patil, "Malicious web pages detection using feature selection techniques and machine learning," *International Journal of High Performance Computing and Networking*, vol. 14, no. 4, pp. 473–488, 2019.

[26] PhishTank, "*Developer Information,*" [Online]. Available: http://phishtank.org/developer_info.php. [Accessed in 28 February 2021].

[27] D. Dua and C. Graff, in *UCI Machine Learning Repository*, University of California, Irvine, CA, USA: School of Information and Computer Science. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Phishing+Websites. [Accessed in 10 January, 2021].

[28] Y. Jiang, G. Tong, H. Yin and N. Xiong, "A pedestrian detection method based on genetic algorithm for optimize XGBoost training parameters," *IEEE Access*, vol. 7, pp. 118310–118321, 2019.