Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2022.020229
Article

Tech Science Press

Enhancing Detection of Malicious URLs Using Boosting and Lexical Features

Mohammad Atrees*, Ashraf Ahmad and Firas Alghanim

Princess Sumaya University for Technology, Amman, 11941, Jordan
*Corresponding Author: Mohammad Atrees. Email: moh20188066@std.psut.edu.jo
Received: 15 May 2021; Accepted: 17 June 2021

Abstract: A malicious URL is a link that is created to spread spams, phishing, malware, ransomware, spyware, etc. A user may download malware that can adversely affect the computer by clicking on an infected URL, or might be convinced to provide confidential information to a fraudulent website causing serious losses. These threats must be identified and handled in a decent time and in an effective way. Detection is traditionally done through the blacklist usage method, which relies on keyword matching with previously known malicious domain names stored in a repository. This method is fast and easy to implement, with the advantage of having low false-positive rates regarding previously recognized malicious URLs. However, this method cannot recognize newly created malicious URLs. To solve this problem, many machine-learning models have been used. In this paper, we introduce an effective machine learning approach that uses an ensemble learner algorithm called AdaBoost (Adaptive Boosting), combined with different algorithms that enhance detection. For datasets filtration, we used CfsSubsetEval technique, which is an algorithm that searches for a subset of features that work well together. Datasets were collected from the UNB repository; divided into four categories: spam, phishing, malware, and defacement URLs; combined with benign URLs, dataset content is based on lexical features. The experimental results indicate that the proposed approach was successful in enhancing the detection accuracy of malicious URLs with less false-positive rates for all experimental algorithms.

Keywords: Malicious URLs; blacklists; machine learning; ensemble learner; adaBoost

1 Introduction

It is all about understanding the threat! One of the most efficient methods of gaining unauthorized access for confidential and private information is social engineering approaches, which are usually carried out by malicious URLs or instant messaging [1,2].

A user could be manipulated to provide sensitive information to a phishing webpage voluntarily, or become a victim of a drive-by-download, ending with a malware infection [3,4].

Organizations train employees on how to identify malicious links to assist in building defenses against these kinds of attacks, but unfortunately, even educated users can get manipulated by visiting non-legitimate websites thinking they are the legitimate ones, especially that nowadays it is easy to clone a website by using cloning tools for misleading and fraud purposes which have led to significant phishing growth [5,6].

Anti-Phishing Working Group (APWG) [7] stated that employees should always be cautious about gift card offers and changes in the payroll account. 65% of Business Email Compromise (BEC) attacks requested gift cards, approximately 20% of BEC attacks demanded diversions of payroll, and 15% demanded direct bank transactions [8,9].

The Webroot threat report 2019 [10] stated that in the year 2018, phishing attacks have grown by 36%, with the number of phishing links detection increasing by 220 percent compared to the year of 2017 [11,12]. Malicious URLs sites manipulate web end-users into assuming that they are secure by adding HTTPS and SSL certificates on a malicious website. It is easy to get SSL certificates from unreliable certificate issuers or a hijacked certificate authority. Fig. 1 below shows the top 10 companies that have been impersonated for phishing attacks [13].

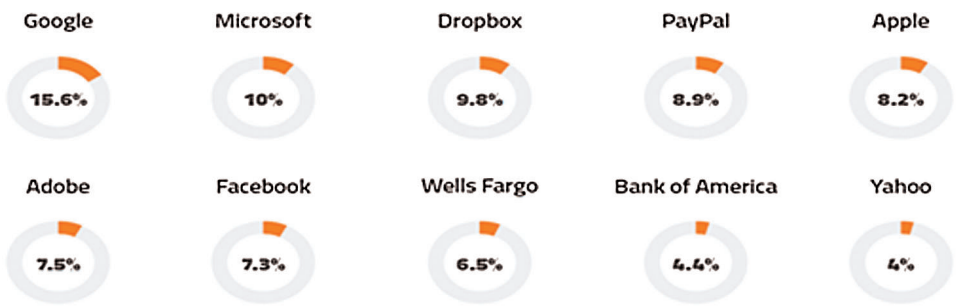

Figure 1: Top 10 impersonated organizations

RSA Quarterly Fraud Report - Q4 2019 [14] stated that "Phishing remains the predominant attack vector used by fraudsters accounting for 60 percent of all fraud attacks"; phishing attacks increase yearly by an average of 54 percent.

The previously mentioned concerns imply the need for using additional protection approaches against malicious URL attacks along with users' awareness; approaches that are capable of identifying malicious websites to prevent users from interacting with [15].

Typically, URL is the first and easiest way to get information about a website. Fig. 2 below show an example of a URL structure, Attackers were aware of the value of this information; they made it clear by using hidden destination URLs that make them hard to detect (by looking like legitimate ones), designing techniques that would distinguish malicious URLs from benign ones is a reasonable solution [16].

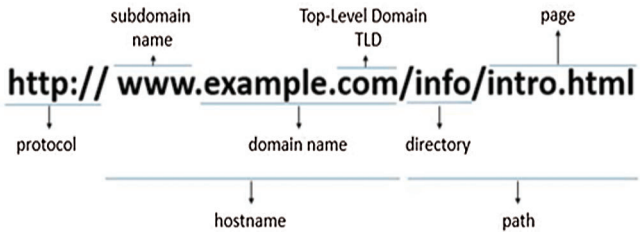

Figure 2: Example of a URL structure

Malicious URLs represent a large portion of cybersecurity threats nowadays. A malicious URL is a link created to spread spams, phishing, traditional malware, Domain generation algorithms (DGA) malware, spyware, etc. You can download malware that will take on your computers by clicking on an infected URL or you can be convinced to provide confidential information (private) on a fraudulent website that may lead to serious losses, for example see Tab. 1.

Table 1: Example of phishing websites URL [17]

Phishing URL	Targeted Brand
http://amazon.co.jp.sdu632id78kjsy4a91kd34kvwsm34.tk/	Amazon
http://93151721113-dot-oppaiwiuwe.rj.r.appspot.com/	Outlook
http://mail.budi1uwoe.wisezk.com/	WhatsApp
http://accounting.affi.online/	PayPal.
http://taylormadegolf.asia/signups/id	Free (ISP)

It is essential that these threats are identified and handled effectively and efficiently. This detection has been traditionally done through blacklist's usage method, which relies on keyword matching that contains previously known malicious URLs domain names (repository). This method is fast and easy to implement, with the advantage of having almost a zero false-positive rate. However, this method cannot recognize newly created malicious URLs. Moreover, it is useless if a malicious website had been wrongly evaluated as a legitimate one. Add to that the nature of the exact match could easily evade the use of URL shortening techniques [18,19].

To solve these problems and to avoid reliance on databases; an approach that is capable of detecting malicious URLs in real-time -without using a blacklist- is needed. In this paper, a machine-learning algorithm is applied. We used AdaBoost ensemble learner combined with other ML algorithms to enhance the detection of malicious URLs [20].

The paper is structured as follows. Section I provides an introduction to the problem studied in this research. Section two introduces the general related topics background. Section three describes the related work for malicious URLs detection. Section four defines the current problem. Section five provides the proposed methodology and dataset details. Section six describes the experimental results. Section seven presents a conclusion and future work suggestions.

2 Background

2.1 Malicious URL Types

Various types of malicious URLs exist, the most popular are phishing, spam, malware (drive-by download), and defacement URLs.

Phishing websites are sites that seek to steal users' private and sensitive information, such as bank card numbers, or user credentials. This is usually done by deceiving the users into thinking they are on a legitimate website. Phishing websites replicate original targeted websites content, which makes it harder for the user to know the difference among them. Spam websites are mainly any sites that try to manipulate search engines to rank higher and to increase their browsing traffic. Drive-by download URLs refer to malware being unintentionally downloaded to the user's computer after visiting malicious URL. Drive-by download

URLs refer to malware being unintentionally downloaded to the user's computer after visiting malicious URL.

2.2 Machine Learning

Machine Learning (ML) allows large data quantities to be analyzed; while delivering faster and more accurate useful results. The general model of Machine Learning is shown in Fig. 3 below. It may also require additional time and resources to prepare the model to start working properly; a process called training.

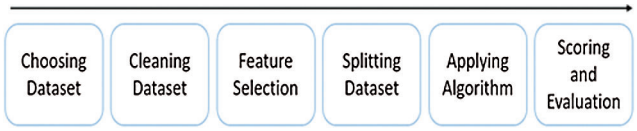

Figure 3: Machine-learning model

Algorithms for Machine Learning are often classified as Supervised or Unsupervised, for the supervised learning, the result or output of the input is known in advance. Common Algorithms: Decision Trees, Support Vector Machines (SVM), Naive Bayes, and Linear Regression. While for the unsupervised learning, the result or output of the input is known in advance. Common Algorithms: Decision Trees, Support Vector Machines (SVM), Naive Bayes, and Linear Regression.

2.2.1 Decision Tree and J48

Decision Tree Algorithms belong to the Supervised Machine Learning family, that is used for classification and regression of problems by applying tree structure form; which breaks existing data into smaller subsets (Nodes and leaves) based on feature values. They are widely used in Machine Learning.

One of most popular decision tree algorithms is J48 (or C4.5). C4.5 is an adaptation of the ID3 algorithm (Iterative Dichotomiser 3), and it was developed by the team of WEKA project [21]. WEKA developers define C4.5 algorithms as "a landmark decision tree program that is probably the machine learning workhorse most widely used in practice to date" [22].

2.2.2 Support Vector Machine (SVM)

SVM is a commonly used ML algorithm, which was developed by Vapnic et al. [23]. SVM implements classification by discovering the hyperplane that got the maximum margin between two classes. Support vectors are the vectors (cases) that define the hyperplane.

2.2.3 Naïve Bayes

Naïve Bayes is a simple method for building classifiers "models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set" [24]. The basic principle for the Naïve Bayes theorem is based on that the value of a particular feature is unrelated to the value to any other feature, given the class variable.

2.2.4 Hoeffding tree

Hoeffding tree is an induction algorithm based on the decision tree. It relies on the fact that we frequently can choose an attribute that is optimal for splitting (from a small sample). The basic idea is derived from the Hoeffding bound [25].

2.2.5 Decision Stump

A decision stump is an ML model that consists of a decision tree with one level. In other words, it is a decision tree with one internal node (root), connected to the terminal nodes (leaves).

2.3 Ensemble Learning

Ensemble learning works by integrating multiple machine learning models into one model to enhance its results. The advantage of using this approach is to get better predictive performance compared to a single algorithm with a higher consistency (avoid overfitting).

The focus in this paper is on the boosting ensemble learner. Boosting is a machine learning technique for classification and regression problems, which is done by using a predefined loss function for error measurement in each step, and error correction in the next one.

One of the most popular ensembles boosting methods is AdaBoost classifier "short for Adaptive Boosting". AdaBoost is an ML meta-algorithm created by Freund and Schapire [26], with the main goal of of it is to enhance the performance of many other forms of machine-learning algorithms. The output of the other machine-learning models (supposed to be weak learners) is transformed into a weighted sum, which reflects the boosted classifier's final output. Adaboost resolves the predictive errors of previous models by enhancing performance, and it works sequentially with many other types of machine learning techniques, Fig. 4 below shows training technique for AdaBoost classifier. The name of AdaBoost in Weka is AdaBoostM1.

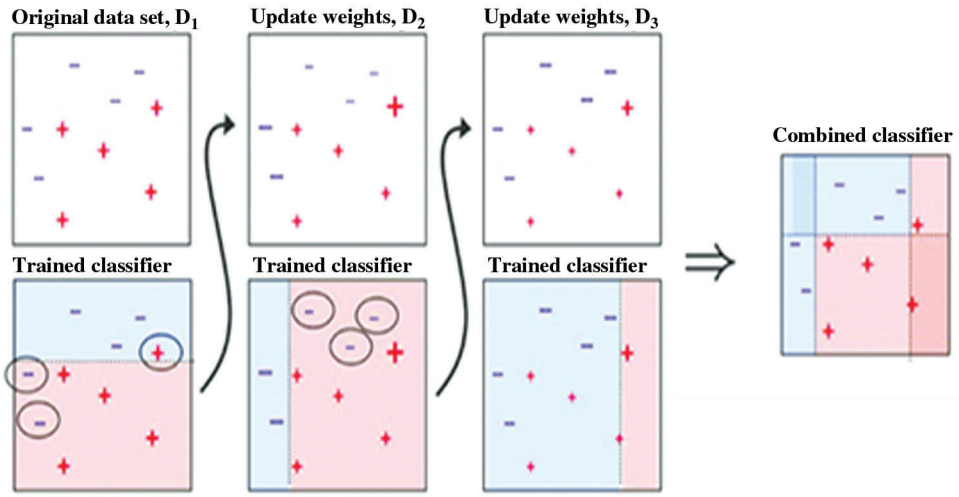

Figure 4: Training of an AdaBoost classifier [27]

2.4 URL Main Component - Feature Analysis

Our focus in this study is on features that are based on readily available URL names (in which if properly used, they can achieve accuracy comparable to full features). Below are some of the main features component explanations [28].

- Length relative features: URL length expanded by inserting additional variable or redirecting URL. Such as urlLen, fileNameLength, Arguments_LongestWordLength, avgpathtokenlen, LongestPathTokenLength.
- Entropy_Domain, Entropy_Extension: Non-legitimate websites frequently embed or replace additional characters into the URL to make it appear like a legitimate one. By inserting characters, the entropy changes more than usual. Alpha entropy is used to classify the randomly generated malicious URLs.

- ldl_getArg: Masquerading in phishing URLs formed by inserting/replacing digits into the letters. The sequence of letters and digits in URL and path is calculated for the detection of these misleading URLs.
- Domain_token_count: Tokens obtained from the URL String (Malicious URLs use multiple domain tokens), Number of tokens in the domains are measured.
- Tld: URL that uses several top-level domains with the same domain name.
- SymbolCount_Domain: A dictionary of delimiters is calculated from a domain (*e.g.*, phishing URLs have more dots compared to benign ones).
- Query_DigitCount: "query part" digits is the number of URLs.

2.5 WEKA

WEKA is a popular machine-learning platform [29]. It includes many machine-learning algorithms that are commonly used. It also enables users to preprocess the big datasets easily before applying ML algorithms.

2.6 Model Evaluation Metrics

The effectiveness of predictive models is typically measured by four main metrics; which are TP "True-Positive: classifier identifies a malicious URL correctly." FP "False-Positive: classifier detects a malicious URL incorrectly." TN "True-Negative: classifier identifies a benign URL correctly.", and FN "False-Negative: classifier identifies a benign URL incorrectly".

Precision is one of the metrics that indicate how trustworthy the model is. The recall is a measure used to explain how many bad URLs the model skipped, which provides a better overview of how well the model detects malicious URLs.

For the success of a machine learning application, it is important to choose the correct evaluation metric of the classifiers. Considering the score only gives an incomplete overview performance of your model and can directly affect its efficacy. Hence, various metrics are used to evaluate the algorithms of Machine Learning, below are the formula of these metrics:

- TPR (True Positive Rate): TP/ (TP+FN).
- FPR (False Positive Rate): FP/ (FP+TN).
- Precision: TP/ (TP+FP).
- Recall: TP/ (TP+FN).
- Accuracy: (TP+TN) / TP+TN+FN+FF.

3 Related Work

Many websites and researchers have introduced different studies and approaches for detecting/blocking malicious URLs. Common approaches use lists of previously known blacklisted URLs. Whenever a user visits a new URL link, a search query is executed within the database, if the URL is blacklisted, it is considered malicious, then an alert generated; otherwise, the URL is considered safe to use. Multiple websites use blacklists to identify malicious URLs such as PhishTank [30], OpenPhish, and DNS-BH [31]. The main drawback of this approach is that detecting new malicious URLs "that aren't in the specified list" would be very difficult.

Sheng et al. [32] investigate the effectiveness of phishing blacklists by testing 191 new phishing URLs on eight anti-phishing tools. He found that 63% of phishing URLs last two hours before they are identified as a malicious link, with only 20% detected at the first hour. This detection approach was ineffective to protect users immediately, as these tools are signature-based URL detection systems, and update at different speeds.

Kan et al. [33] introduce a faster approach for webpage classification than the typical ones, which is done by URL segmentation into meaningful extracted features. The URL-based approach resulted in better performance rates than full text and link-based approaches.

Machine learning was used in various methods for the detection of malicious URLs. Ntoulas et al. [34] used content-based analysis to detect spam webpages, like the number of words on the page and page title, the average length of words, etc. The effectiveness of these contents is evaluated by the CS4.5 algorithm and boosting ensemble learning.

Basnet et al. [35] proposed a machine learning approach for phishing attack detection; they have used many algorithms such as Neural Networks, Support Vector Machines (SVMs), Biased Support Vector Machines (BSVMs), and K-Means. They used 16 structural features, features that allow for spam detection such as IP-based URL, Age of Domain Name, and Number of Domain and SubDomains, etc. SVM achieved the highest detection results.

Nezhad et al. [36], proposed a way of analyzing features set, including JavaScript, HTML, and XSS attacks. The evaluation was done by machine learning algorithms such as ANN, Naïve Bayes, SVM, CS4.5, and KNN. CS4.5 algorithm shows the best performance accuracy with a 97.61% rate.

Fette et al. [37] proposed an approach and named it PILFER to classify phishing messages. Their classifiers check the properties of URLs content within a message such as the number of URLs, number of dots in a URL, and Age of linked to domain names, etc. they consider features of the email structure and content. For implementation, they used Random Forest, SVM, and Naïve Bayes machine-learning algorithms.

In this paper, we use the AdaBoost ensemble learner combined with ML algorithms for the sake of reaching better outperforming results.

4 Problem

Blacklisted methods cannot recognize newly created malicious URLs, and they are useless if a malicious website has been wrongly evaluated as a legitimate one. Add to that the nature of the exact match could be easily evaded with the use of URL shortening techniques.

Attackers are using the Internet as their tool to attack users' devices to obtain private information using fraudulent URLs. Add to that, nowadays it is easy to clone a website by using cloning tools for misleading and fraud purposes, even trained users can be manipulated by these types of websites, all of these issues and other advanced malicious techniques led to significant phishing growth. We can detect such potential risks and act against them.

Previous concerns raise the need for an approach that is capable of improving the performance of detecting malicious URLs in real-time without the need for database dependency as we do in the blacklisting method. Our approach uses a boosting technique called AdaBoost (Adaptive Boosting), combined with different machine learning algorithms.

5 Proposed Model

Our target is to build an enhanced machine learning approach that identifies malicious URLs with high accuracy, and without having to download potentially harmful page content. Within this experiment, we will examine out whether the performance of standard machine learning algorithms can be enhanced by using a boosting technique.

Evaluating the proposed model is done by comparing the results before and after applying our proposed model. Fig. 5 below illustrates the proposed methodology.

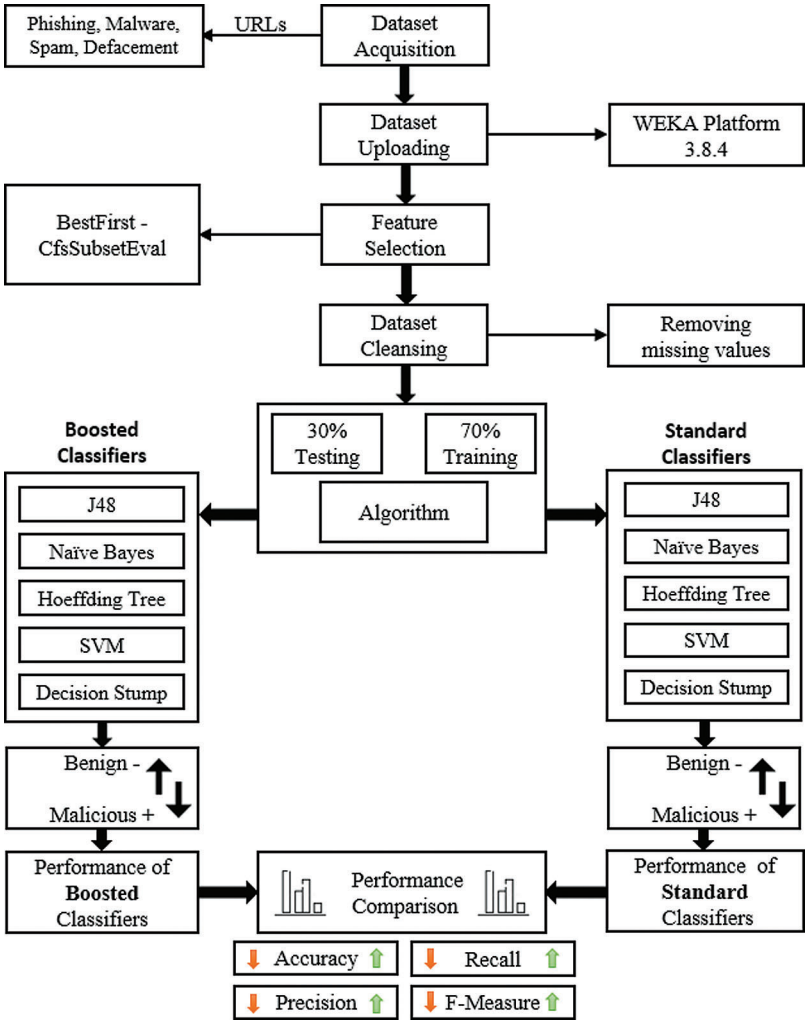

Figure 5: Proposed model

5.1 Experimental Environment

The experiment is conducted using windows 10 OS machine with Intel Core i7-8550U @1.8 GHz Desktop Processor, 8 GB memory. For model implementation and evaluation, Weka 3.8.4 platform is used.

5.2 Using Machine Learning in Malicious URL Detection

The user will enter a website address; our detection model will classify the entered URL to a legitimate website or malicious website.

In the first phase of the proposed approach, we used the following ML algorithms: J48, Naïve Bayes, Hoeffding tree, SVM, and Decision Stump algorithms. For the second phase, we combined previously mentioned algorithms with AdaBoost algorithm, seeking for detection enhancement.

5.3 Data Set Acquisition

Different types of malicious datasets were collected from the University of New Brunswick UNB repository [38], a publicly accessible source. Dataset features were collected based on URL names.

Datasets are available in four separate single-class categories: Phishing, Malware, Spam, and Defacement URLs, combined with benign URLs obtained from Alexa top websites. Different selected features available in these datasets.

From UNB listed datasets we chose the BestFirst CSV files, as they are filtered by CfsSubsetEval technique, which is an algorithm that searches for a subset of features that work well together (have a low correlation between the features and a high correlation to the target label), This results in a better performing model, that is easier to understand and run faster.

For authenticity purposes, we filtered the original datasets that contain the full features (80 attributes) by using the CfsSubsetEval technique and compared it with the previously chosen datasets, which returned the same results as the chosen ones. Figs. 6–9 below show filtering results for each type of dataset.

```
Selected attributes: 2,7,20,21,24,27,35,44,51,59,60,69,75 : 13
                     domain_token_count
                     tld
                     urlLen
                     domainlength
                     fileNameLen
                     pathurlRatio
                     NumberofDotsinURL
                     Query_DigitCount
                     LongestPathTokenLength
                     delimeter_Domain
                     delimeter_path
                     SymbolCount_Domain
                     Entropy_Domain
```

Figure 6: Phishing dataset features

```
Selected attributes: 2,7,20,29,35,63,69,75,78 : 9
                     domain_token_count
                     tld
                     urlLen
                     argDomanRatio
                     NumberofDotsinURL
                     NumberRate_Domain
                     SymbolCount_Domain
                     Entropy_Domain
                     Entropy_Extension
```

Figure 7: Malware dataset features

```
Selected attributes: 2,7,14,35,60,69 : 6
                     domain_token_count
                     tld
                     ldl_getArg
                     NumberofDotsinURL
                     delimeter_path
                     SymbolCount_Domain
```

Figure 8: Spam dataset features

```
Selected attributes: 2,6,7,28,35,55,58,59,60,64,69,75 : 12
                     domain_token_count
                     avgpathtokenlen
                     tld
                     ArgUrlRatio
                     NumberofDotsinURL
                     Arguments_LongestWordLength
                     spcharUrl
                     delimeter_Domain
                     delimeter_path
                     NumberRate_DirectoryName
                     SymbolCount_Domain
                     Entropy_Domain
```

Figure 9: Defacement dataset features

Details of each dataset are mentioned below, details regarding the number of malicious and benign URLs, available features, and URL data sources.

5.3.1 Phishing URLs Dataset

A 15367 benign and phishing URLs were obtained from the OpenPhish repository. Dataset available in CSV format contains 7586 phishing URLs and 7781 benign URLs. Thirteen selected features are available in this dataset and one for class type.

5.3.2 Malware URLs Dataset

Around 14,487 benign and Malware websites were obtained from DNS-BH which is a website that contains several malware URLs updated lists. Dataset available in CSV format contains 6707 malware URLs with 7780 benign URLs, 9 selected features are available in this dataset and one for class type.

5.3.3 Spam URLs Dataset

A 14,479 benign and spam URLs were obtained from the WEBSPAM-UK2007 dataset. Dataset available in CSV format contains 6698 spam URLs with 7781 benign URLs, 6 selected features are available in this dataset and one for class type.

5.3.4 Defacement URLs Dataset

A 15,711 benign and defacement URLs. Defacement website referred to URLs that host fraud or hidden malicious links, they are an Alexa trusted websites ranked list. Dataset available in CSV format contains 7930 defacement URLs with 7780 benign URLs, 12 selected features are available in this dataset and one for class type.

The following Tab. 2 summarizes the number of malicious and benign URLs in each dataset.

Table 2: Datasets URLs count

URLs Dataset	Malicious URLs	Benign URLs	Total
Phishing	7586 (49%)	7781 (51%)	15367
Malware	6707 (46%)	7780 (54%)	14487
Spam	6698 (46%)	7781 (54%)	14479
Defacement	7930 (50%)	7780 (50%)	15711

5.4 Dataset Uploading

We uploaded phishing, malware, spam, and defacement datasets to Weka software 3.8.4 version.

5.5 Dataset Cleansing

Our first step is the dataset cleansing process to check and remove any missing data (values).

5.6 Dataset Splitting

Next, for training and testing purposes; we divide the dataset into two parts: Training Dataset (70 percent split ratio) and test dataset (30 percent split ratio). Tab. 4 below shows the used splitting ratio.

5.7 Training Phase & ML Algorithm Selection

In the first phase, we use standard J48, Naïve Bayes, Hoeffding tree, SVM, and Decision Stump algorithms. For the second phase, we add AdaBoost learner (with its default configurations) to the previously mentioned classifiers. Then we connect the trained data set with the proposed approach model. The next step is choosing our column target for future prediction (Class column). After training our proposed model with the training data, we proceed to the testing phase. To evaluate our model we used the previously discussed metrics.

6 Results

This section presents the results of the carried out experiment based on the proposed model. True Positive (TP) represents the correctly identified malicious URLs, while True Negative (TN) is the correctly identified benign URLs. The event of False Positive (FP), is when a URL is predicted to be malicious but it is benign, while False Negative (FN) is when a URL is predicted to be benign but it is malicious. By combining these categories, we create metrics (confusion matrix) that help to evaluate our proposed model performance before and after implementation. The following measures are used to evaluate the proposed model: accuracy, precision, recall, f-measure, TPR, FPR, and confusion matrix.

To verify the effectiveness of our proposed model, we compared classifiers' performance before and after applying AdaBoost learner. As shown in the following tables, the use of AdaBoost improved the overall performance of J48, Naïve Bayes, Hoeffding tree, SVM, and Decision Stump classifiers on all tested datasets at different levels.

The best standard algorithm that shows the highest overall performance through all collected datasets was the J48 algorithm, while the Decision Stump algorithm got the lowest overall performance. Fig. 10

shows accuracy performance results for standard algorithms and the results after implementing AdaBoost learner on all datasets; we noticed an overall significant performance improvement by decision stump, Hoeffding tree, and Naïve Bayes algorithm.

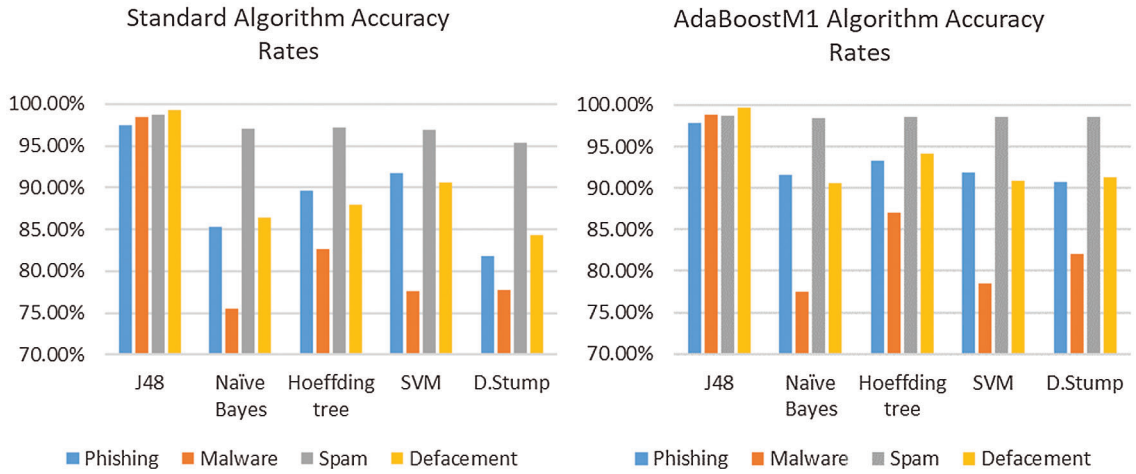

Figure 10: Accuracy comparison among standard ML algorithms and among Boosted ML models

Decision Stump algorithm got the highest overall detection performance improvement on all datasets, by boosting the detection rate with an increase of 8.94% on phishing, 4.19% on malware, 3.23% on spam, and 6.96% on defacement URLs, with no differences regarding execution time with boosted one.

Boosted Naïve Bayes finished with an increase of 6.19% on phishing, 1.98% on malware, 1.50% on spam, and 4.25% on defacement URLs. Boosted Hoeffding tree finished with an increase of 3.67% on phishing, 4.33% on malware, 1.42% on spam, and 6.11% on defacement URLs. Boosted SVM got results with an increase of 0.11% on phishing, 0.90% on malware, 1.70% on spam, and 0.15% on defacement URLs. Finally, boosted J48 finished with an increase of 0.41% on phishing, 0.44% on malware, 0.07% on spam, and 0.45% on defacement URLs. Fig. 11 shows the detection increase rates after implementing AdaBoost learners. Fig. 12 shows TPR and FPR performance for Decision Stump before and after applying AdaBoost learner.

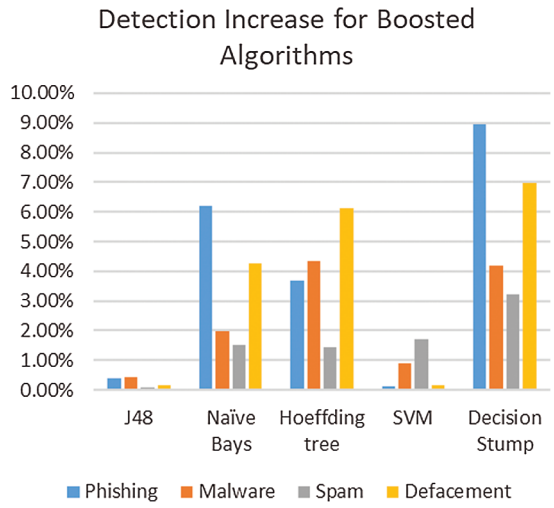

Figure 11: Detection increase rates after applying the boosting technique

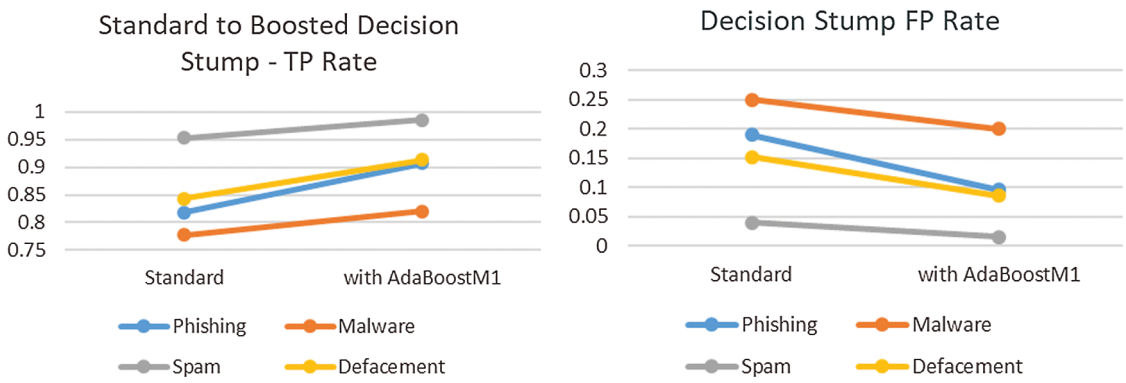

Figure 12: Detection TP and FP rates after applying the Adaboost

Although the Decision Stump algorithm got the highest performance improvement changes, its accuracy is still less than the requirements. As shown in Fig. 13 below, the best performance after applying the AdaBoost technique is the J48 algorithm with the highest accuracy rate among our experimented algorithms, with an accuracy rate of 97.89% on phishing, 98.87% on malware, 98.73% on spam, and 99.72% on defacement URLs.

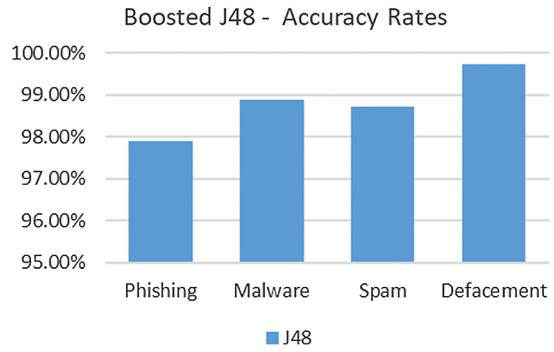

Figure 13: Boosted J48 accuracy rates on malicious dataset

6.1 J48 Algorithm

The classifier achieves an overall average accuracy rate of 98.46% without AdaBoost, and 98.80% with AdaBoost. This classifier performs well on all datasets. The results are summarized in Tab. 3, which shows the precision, recall and accuracy performance values of J48 classifier on all datasets before and after applying AdaBoost learner.

6.2 Naïve Bayes Algorithm

The classifier achieves an overall average accuracy rate of 86.05% without AdaBoost, and 89.53% with AdaBoost. This classifier does not perform well on the malware dataset even after applying AdaBoost. The results are summarized in Tab. 4. It shows the precision, recall and accuracy performance of Naïve Bayes classifier on all datasets, before and after applying AdaBoost.

Table 3: J48 Performance comparison in terms of precision, recall and accuracy

URLs Dataset	J48 Standard			J48 + Boosted			Accuracy Changes%
	Precision	Recall	Accuracy	Precision	Recall	Accuracy	
Phishing	0.975	0.975	97.48 %	0.979	0.979	97.89 %	0.41
Malware	0.984	0.984	98.43 %	0.989	0.989	98.87 %	0.44
Spam	0.987	0.987	98.66 %	0.987	0.987	98.73 %	0.07
Defacement	0.993	0.993	99.27 %	0.997	0.997	99.72 %	0.45
Average			98.46%			98.80%	0.34

Table 4: Naïve bayes performance comparison in terms of precision, recall and accuracy

URLs Dataset	Naive Bayes			Naive Bayes + Boosted			Accuracy Changes%
	Precision	Recall	Accuracy	Precision	Recall	Accuracy	
Phishing	0.861	0.854	85.35 %	0.916	0.915	91.54 %	6.19
Malware	0.763	0.755	75.51 %	0.779	0.775	77.49 %	1.98
Spam	0.971	0.970	96.98 %	0.985	0.985	98.48 %	1.50
Defacement	0.866	0.864	86.37 %	0.906	0.906	90.62 %	4.25
Average			86.05%			89.53%	3.48

6.3 Hoeffding Tree Algorithm

The classifier achieves an overall average accuracy rate of 89.38% without AdaBoost, and 93.26% with AdaBoost. This classifier performs well on all datasets, especially on the spam dataset. The results are summarized in Tab. 5, which shows the precision, recall and accuracy performance of the Hoeffding tree classifier on all datasets, before and after applying AdaBoost learner.

Table 5: Hoeffding tree performance comparison in terms of precision, recall and accuracy

URLs Dataset	Hoeffding tree Standard			Hoeffding tree + Boosted			Accuracy Changes%
	Precision	Recall	Accuracy	Precision	Recall	Accuracy	
Phishing	0.897	0.897	89.69 %	0.934	0.934	93.36 %	3.67
Malware	0.827	0.827	82.71 %	0.874	0.870	87.04 %	4.33
Spam	0.973	0.971	97.12 %	0.986	0.985	98.54 %	1.42
Defacement	0.900	0.880	88.01 %	0.941	0.941	94.12 %	6.11
Average			89.38%			93.26%	3.88

6.4 SVM Algorithm

The classifier achieves an overall average accuracy rate of 89.20% without AdaBoost, and 89.91% with AdaBoost. This classifier does not perform well on the malware dataset even after applying the AdaBoost method. The results summarized in Tab. 6 show the precision, recall and accuracy performance of the SVM classifier on all datasets, before and after applying AdaBoost.

Table 6: SVM performance comparison in terms of precision, recall and accuracy

URLs Dataset	SVM Standard			SVM + Boosted			Accuracy Changes%
	Precision	Recall	Accuracy	Precision	Recall	Accuracy	
Phishing	0.918	0.918	91.75 %	0.919	0.919	91.86 %	0.11
Malware	0.794	0.776	77.56 %	0.789	0.785	78.46 %	0.90
Spam	0.970	0.968	96.84 %	0.986	0.985	98.54 %	1.70
Defacement	0.907	0.907	90.66 %	0.908	0.908	90.81 %	0.15
Average			89.20%			89.91%	0.71

6.5 Decision Stump Algorithm

The classifier achieves an overall accuracy rate of 84.81% without AdaBoost, and 90.64% with AdaBoost. This classifier does not perform well on the malware dataset even after applying the AdaBoost method. The results are summarized in Tab. 7. It shows the precision, recall and accuracy performance of the Decision Stump classifier on all datasets, before and after applying AdaBoost.

Table 7: Decision stump performance comparison in terms of precision, recall and accuracy

URLs Dataset	Decision Stump Standard			Decision Stump + Boosted			Accuracy Changes%
	Precision	Recall	Accuracy	Precision	Recall	Accuracy	
Phishing	0.830	0.818	81.77 %	0.910	0.907	90.71 %	8.94
Malware	0.802	0.778	77.79 %	0.832	0.820	81.98 %	4.19
Spam	0.958	0.953	95.34 %	0.986	0.986	98.57 %	3.23
Defacement	0.856	0.843	84.34 %	0.913	0.913	91.30 %	6.96
Average			84.81%			90.64%	5.83

7 Conclusion and Future Work

We have discussed in this paper the challenges of detecting malicious URLs content using traditional methods, and how machine learning helped to address these challenges by providing effective models that capture a larger distribution of malicious URLs

The main experimental work of this paper was to determine if the machine-learning approach combined with an ensemble learning technique, AdaBoost, can improve standard algorithms performance.

Using the proposed technique combined with J48, Naïve Bayes, Hoeff.tree, SVM, and Decision Stump algorithms, led to boost detection of malicious URLs in terms of accuracy and True Positive rates on different datasets.

From our proposed model results, we corroborate the benefit of using AdaBoost ensemble learners with weak and strong algorithms, which was done by achieving higher detection rates and lower False Positive rates.

We also noticed that some classifiers do not perform well on malware dataset, as they achieved accuracy rates less than 80%, we can take the benefit of exploring/adding more relative features on malware dataset to improve classifiers performance.

Each algorithm has its pros and cons; hence, to make the proposed model detection work better, we used AdaBoost combined with other algorithms. To take the full advantages from such an approach, we will explore more algorithms, with different datasets, and try to implement other ensemble techniques like bagging, stacking, and voting ensemble learners.

Although there are several methods for malicious URLs detection, so far no single solution is ideal or sufficient to assure 100% system security. Only a combination of additional system security measures can give us additional security protection.

Funding Statement: The authors received no specific funding for this study.

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

References

[1] H. Aldawood and G. Skinner, "Reviewing cyber security social engineering training and awareness programs—Pitfalls and ongoing issues," *Future Internet*, vol. 11, no. 3, pp. 73, 2019.

[2] N. Patel, "Social engineering as an evolutionary threat to information security in healthcare organizations," *Jurnal Administrasi Kesehatan Indonesia*, vol. 8, no. 1, pp. 56, 2020.

[3] A. Surwade, "Phishing e-mail is an increasing menace," *International Journal of Information Technology*, vol. 12, no. 2, pp. 611–617, 2020.

[4] B. Cerda and S. Yuan, "A Study of anti-phishing methodologies and phishing detection algorithms," in *Proc. of the Int. Conf. on Security and Management (SAM)*, Las Vegas, USA, pp. 79–83, 2019.

[5] J. Oravec, "Emerging cyber hygiene practices for the Internet of Things (IoT): Professional issues in consulting clients and educating users on IoT privacy and security," in *IEEE Int. Professional Communication Conference (ProComm)*, Wisconsin, USA, pp. 1–5, 2017.

[6] S. Chaudhary, "The use of usable security and security education to fight phishing attacks," 2016.

[7] D. Glăvan, C. Răcuciu, R. Moinescu and S. Eftimie, "Detection of phishing attacks using the anti-phishing framework," *Scientific Bulletin" Mircea cel Batran" Naval Academy*, vol. 23, no. 1, pp. 208–212, 2020.

[8] M. Somesha, A. Pais, R. Rao and V. Rathour, "Efficient deep learning techniques for the detection of phishing websites," *Sādhanā*, vol. 45, no. 1, pp. 3851, 2020.

[9] T. Sujithra, N. Dwivedi and A. Utakarsha, "Detection of phishing websites using deep learning and machine learning," *Journal of Critical Reviews*, vol. 7, pp. 1027–1032, 2020.

[10] H. Lonas, "Webroot Threat Report," pp.18, 2019. [Online]. Available: https://www.webroot.com/download_file/3181.

[11] C. Hadnagy, *Phishing as-a-service (Phaas) used to increase corporate security awareness*. ed: Google Patents, 2017.

[12] T. Chin, K. Xiong and C. hu, "Phishlimiter: A phishing detection and mitigation approach using software-defined networking," *IEEE Access*, vol. 6, no. 99, pp. 42516–42531, 2018.

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

[14] C. Tan and K. Chiew, "Phishing webpage detection using weighted URL tokens for identity keywords retrieval," *9th Int. Conf. on Robotic, Vision, Signal Processing and Power Applications*, vol. 398, pp. 133–139, 2017.

[15] C. Tan, K. Chiew, K. Yong, S. Sze, J. Abdullah *et al.,* "A graph-theoretic approach for the detection of phishing webpages," *Computers & Security*, vol. 95, pp. 101793, 2020.

[16] P. Nguyen, D. Ha, A. Jaafari, H. Nguyen, T. Phong *et al.,* "Groundwater potential mapping combining artificial neural network and real AdaBoost ensemble technique: The DakNong province case-study," *Vietnam International Journal of Environmental Research and Public Health*, vol. 17, no. 7, pp. 2473, 2020.

[17] R. Rao, T. Vaishnavi and A. Pais, "CatchPhish: Detection of phishing websites by inspecting URLs," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 2, pp. 813–825, 2020.

[18] R. Rao and A. 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.

[19] J. Hong, T. Kim, J. Liu and N. Park, "Phishing url detection with lexical features and blacklisted domains," in *Adaptive Autonomous Secure Cyber Systems*. ed: Springer, pp. 253–267, 2020.

[20] H. Jiang, W. Zheng, L. Luo and Y. Dong, "A two-stage minimax concave penalty based method in pruned AdaBoost ensemble," *Applied Soft Computing*, vol. 83, no. 1, pp. 105674, 2019.

[21] T. N. Shah, M. Zakir Khan, M. Ali, B. Khan and N. Idress, "J48, ID3, Decision stump and random forest: A comparative study," *University of Swabi Journal (USJ)*, vol. 2, no. 1, pp. 1–6, 2018.

[22] G. Kumari, M. Kumar and Dr. A. Sowjanya, "Accident severity prediction on narrative reports using text-mining," *Journal of Information and Computational Science*, vol. 10, no. 7, pp. 563–570, 2020.

[23] S. Heddam, R. Adnan, Z. Liang, M. Kermani, O. Kisi *et al.,* "Least square support vector machine and multivariate adaptive regression splines for streamflow prediction in mountainous basin using hydro-meteorological data as inputs," *Journal of Hydrology*, vol. 586, no. 123981, pp. 124371, 2020.

[24] S. Chen, G. Webb, L. Liu and X. Ma, "A novel selective naïve Bayes algorithm," *Knowledge-Based Systems*, vol. 192, no. 1, pp. 105361, 2020.

[25] E. García-Martín, A. Bifet and N. Lavesson, "Energy modeling of Hoeffding tree ensembles," *Intelligent Data Analysis*, vol. 25, no. 1, pp. 81–104, 2021.

[26] Y. Yue and Y. Yang, "Improved AdaBoost classifier for sports scene detection in videos: From data extraction to image understanding," in *2020 Int. Conf. on Inventive Computation Technologies (ICICT)*, Kerala, India, pp. 1–4, 2020.

[27] D. Tang, L. Tang, R. Dai and J. Chen, "MF-Adaboost: LDoS attack detection based on multi-features and improved Adaboost," *Future Generation Computer Systems*, vol. 106, no. 7, pp. 347–359, 2020.

[28] M. Rajab, "Visualisation model based on phishing features," *Journal of Information & Knowledge Management*, vol. 18, no. 01, pp. 1950010, 2019.

[29] S. Hussain, M. Jibril, F. Ishaq and A. Yakubu, "Performance evaluation of various data mining algorithms on road traffic accident dataset," in *Information and Communication Technology for Intelligent Systems*. ed: Springer, pp. 67–78, 2019.

[30] R. PhishTank, "PhishTank," 2019, Available: https://www.phishtank.com.

[31] H. Khan, Q. Niyaz, V. Devabhaktuni and S. Guo, "Identifying generic features for malicious URL detection system," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conf. (UEMCON)*, New York, USA, pp. 0347–0352, 2019.

[32] S. Sheng, B. Wardman, G. Warner and L. Cranor, "An empirical analysis of phishing blacklists," 2009.

[33] M. Kan and H. O. Thi, "Fast webpage classification using URL features," in *Proc. of the 14th ACM int. conf. on Information and knowledge management*, New York, USA, pp. 325–326, 2005.

[34] A. Ntoulas, M. Najork, M. Manasee and D. Fetterly, "Detecting spam web pages through content analysis," in *Proc. of the 15th int. conf. on World Wide Web*, New York, USA, pp. 83–92, 2006.

[35] R. Basnet, S. Mukkamala and A. Sung, "Detection of phishing attacks: A machine learning approach," in *Soft computing applications in industry*. ed: Springer, pp. 373–383, 2008.

[36] J. Nezhad, M. Jahan, M. Tayarani and Z. Sadrnezhad, "Analyzing new features of infected web content in detection of malicious web pages," *ISeCure-The ISC International Journal of Information Security*, vol. 9, no. 2, pp. 161–181, 2017.

[37] I. Fette, N. Sadeh and A. Tomasic, "Learning to detect phishing emails," in *Proc. of the 16th int. conf. on World Wide Web*, New York, USA, pp. 649–656, 2007.

[38] U, O. N. Brunswick(2016, 21/8) www.unb.ca/cic/datasets/url-2016.html.